Michael Seaholm
12-19-2012
CS 760 Section 1
Project Report

A Comparison of Learning Methods for Spam Classification

Abstract: Given its adaptive and pervasive nature, spam is often classified by means of machine learning techniques, which have the ability to finely discern the difference between spam and legitimate messages in an automated fashion. Popular applications for spam classification typically incorporate some sort of Bayesian mechanism in order to classify spam; however, there is no reason to think that other classification methods cannot be used for the same purpose. For this project we implemented three spam classifiers following different learning approaches and evaluated them with regards to a sample spam classification task. We then discuss our interpretation of the results with regards to the decision boundary and learning model involved with each classifier as well as how each classifier holds up against adaptations in spam over time.

1. Introduction

Anyone with an e-mail address is probably well aware of the presence of junk e-mail, or spam, and its impact on Internet users everywhere. Over 90 trillion spam e-mails are thought to have circulated in 2010 [1], and the figures for the years to come are unlikely to be any less staggering. However, most end-users do not have to deal firsthand with the majority of this spam owing to the intervention of filtering programs which separate spam from legitimate mail, also called ham. Such spam filters or classifiers are integral to keeping users' e-mail accounts from becoming overburdened with spam, thereby preventing spam from becoming an active impediment to general e-mail access as opposed to a comparatively minor nuisance.

Many spam classifiers use Bayesian statistics in order to make inferences regarding the class of incoming e-mails. In particular, the program SpamAssassin, which is maintained by Apache Software Foundation and sees widespread public and private use, includes a naïve Bayes classifier at its core which adapts with incoming spam over time [2]. Although naïve Bayes approaches are especially prevalent as an option for spam filtering, it is reasonable to think that other machine learning techniques could be applied to the task of spam classification. Toward that end, we implemented three spam classifiers - one each for the naïve Bayes, logistic regression, and multilayer neural network paradigms - and made an evaluative comparison between each given the task of classifying spam from a selected data set.

The rationale behind this particular choice of classifiers is as follows. Because of its general utility in performing this task, naïve Bayes is a natural choice for one of the classifiers, as it serves as a baseline by which the other classifiers can be evaluated. The selection of logistic regression as one of the implementations stems from the equivalence in functional form shared between this approach and naïve Bayes [3]; more specifically, it allows for a direct comparison between generative and discriminative learning models as applied to the spam classification task. It should be noted that both of these approaches separate instances using a linear decision boundary in the feature space [3]. This is why a multilayer neural network was also implemented for this task, as its ability to learn non-linear boundaries [4] may provide insight into the problem of classifying spam.

Going into this project, we sought to address a few questions motivated by our choice of classifiers and the task of spam classification. First, how does performance in spam classification vary between discriminative and generative approaches? Similarly, how is performance affected

by the form of the decision boundary in the feature space - linear or non-linear - that the classifier is able to learn? The extent to which performance is influenced by each of these factors is explored more fully during the evaluation phase of the project, in which we consider not only basic differences in accuracy over the sample data set but also a measure of how well each classifier adapts to more recent ham and spam messages after having trained on older instances.

2. Approach

2.1. The Classifiers

As stated previously, this project dealt with the implementation of three spam classifiers: a naïve Bayes network, a logistic regressor, and a multilayer neural network. Each was implemented in such a way as to ensure that no one classifier had an *a priori* advantage over the others during the evaluation phase; that is, the basic principles for each approach were used in building each classifier while avoiding any classifier-specific optimizations which might significantly increase accuracy over the chosen data set. Certain design decisions were made on a per-classifier basis during the implementation phase, which are enumerated in the description for each classifier below.

The naïve Bayes classifier consists of a naïve Bayes network in which the primary parent node represents the class label for a given instance and whose child nodes represent the relevant features. The network is trained by using inference by enumeration over a training set and the resulting probability estimates are smoothed using Laplacian pseudocounts [5]. Instances from the test set are subsequently classified by finding the probability of the class label given the features using Bayes' rule to make the computation feasible [5]. This follows from the standard form of a naïve Bayes classifier, and for our purposes no further additions were required.

In contrast to the naïve Bayes network, the logistic regression classifier takes the form of a series of input units connected by weighted edges to a sigmoid output, in essentially the same layout as a perceptron. The weights for the logistic regressor are trained by means of gradient ascent to maximize the sum of log probabilities for the class label given the features [6]. In this implementation, the regressor uses stochastic gradient ascent in order to promote faster convergence, which was not deemed to affect accuracy unfairly but rather increase efficiency to facilitate testing. Output from the regressor is calculated by taking the sigmoid function applied to the sum of products of weights and input values for each instance [6]. This constitutes the logistic regression classifier that was created for this project, serving as the discriminative counterpart to naïve Bayes.

The multilayer neural network classifier is similar in form to the logistic regressor previously described, although more complex in operation. An additional layer of hidden units was put between the input and output units which would normally be present in a simple perceptron. Each input unit is fully connected to the units in this intermediate hidden layer, and each hidden unit points to a single sigmoid output. The number of hidden units selected for this task is roughly half of the sum of the number of input and output units, as per the recommendation of various sources [7] [8]. Training for this network involves updating the weights between the various units using backpropagation, and more specifically with stochastic gradient descent to minimize the sum squared error of the network output [4]. In contrast to the other two classifiers, this neural network is capable of learning a non-linear decision boundary, which may have a noticeable impact during evaluation.

2.2. The Data Set

Because the data set used in the evaluation phase was essentially "home grown", much of the initial effort for this project was centered around retrieving the necessary data and converting it into a useful format for use by the classifiers. The original data set consisted of one thousand e-mails from the author's workplace inbox, chosen randomly and divided evenly between ham and spam. From there, a small script removed the metadata from each message and then scanned them for relevant textual features. This produced a series of continuous values for each instance, with each value corresponding to a specific feature, along with appropriate class labels. It should be noted that these feature values are later discretized by the classifiers themselves, as it was found that each approach saw more benefit as compared to using the continuous inputs directly.

The original set of features considered for this project was taken directly from the Spambase data set description available through the University of California Irvine Machine Learning Repository [9]. This set consists of 57 features whose values correspond to percentages of matches to specific strings (words, individual characters, and sequences of capital letters) that may be present in a given text [9]. However, since the original Spambase data set is over ten years old, we found that the features given in the provided description did not properly represent ham and spam instances to the point where they could be reliably distinguished from one another. As such, the matching strings which are used to derive the feature values have been modified through an iterative process meant to ensure that a reasonably up-to-date feature set would be used during classification.

To determine which matching strings to keep from the original feature set described by Spambase, we used forward selection over all three classifiers to find all matching string features which produced accurate results when taken collectively. This greatly reduced the size of the feature set, and so to find additional good matching string features, we looked at the most frequent strings occurring in the ham and spam messages in a training subset of our original data set. From there, we used backward elimination over our classifiers to weed out unimportant features as necessary. We repeated this process until we obtained roughly as many features as there were in the original Spambase data set. In all, there are 62 features in the final data set used for this project.

From this new feature set description, we reprocessed the individual e-mails comprising our original data set and output their corresponding feature values and class labels into a file in the ARFF format, which the classifiers can then read. This forms the basis for the subsequent evaluation of these classifiers.

3. Evaluation

After we had finished implementing the three classifiers chosen for this project and preparing the data set as described above, we began an extensive comparison of each classifier given our spam classification task. Going into the evaluation phase, we looked back at our questions regarding how performance might be affected by the class of model (discriminative or generative) and form of the learnable decision boundary (linear or non-linear) for each classifier and made a few predictions as to what we expected each of these factors to accomplish in our evaluation. Our primary motivation for the project was brought about by the relative ubiquity enjoyed by the naïve Bayes approach in classifying spam online, and so we assumed that this may be due to an inherent advantage that generative models may have in performing this task. As

well, we reasoned that a non-linear classifier would presumably do as well at handling linearly separable instances as a linear classifier since it is in general a more expressive learner. As such, we sought to verify the following hypotheses:

1. The naïve Bayes classifier, using a generative model, will on average outperform the discriminative classifiers in terms of accuracy given the spam classification tasks used in these experiments.
2. The multilayer neural network, capable of learning a non-linear decision boundary, will on average outperform the logistic regressor and its linear decision boundary in terms of accuracy given the same spam classification tasks.

As we will show, certain aspects of our hypotheses were sound, both in general and under specific conditions, and some were not. Details toward this end will become more apparent once we have described the evaluation phase in sufficient detail.

For the initial experiment, each classifier was run over the thousand-instance data set previously described, for which half of the instances were ham and the other half spam. This experiment sought to judge each classifier over the thousand instances using stratified *k*-fold cross validation with *k* set to 10 as a means of ensuring a more robust estimation of overall accuracy. The choice of learning rate and number of training epochs for the multilayer neural network and logistic regressor, as well as the number of bins used by each individual classifier when discretizing the continuous feature values given in the data set, were selected empirically to maximize per-classifier accuracy over this data set while keeping running time under a minute for each trial. The resulting accuracy of each classifier over ten trials is given in Figure 1.

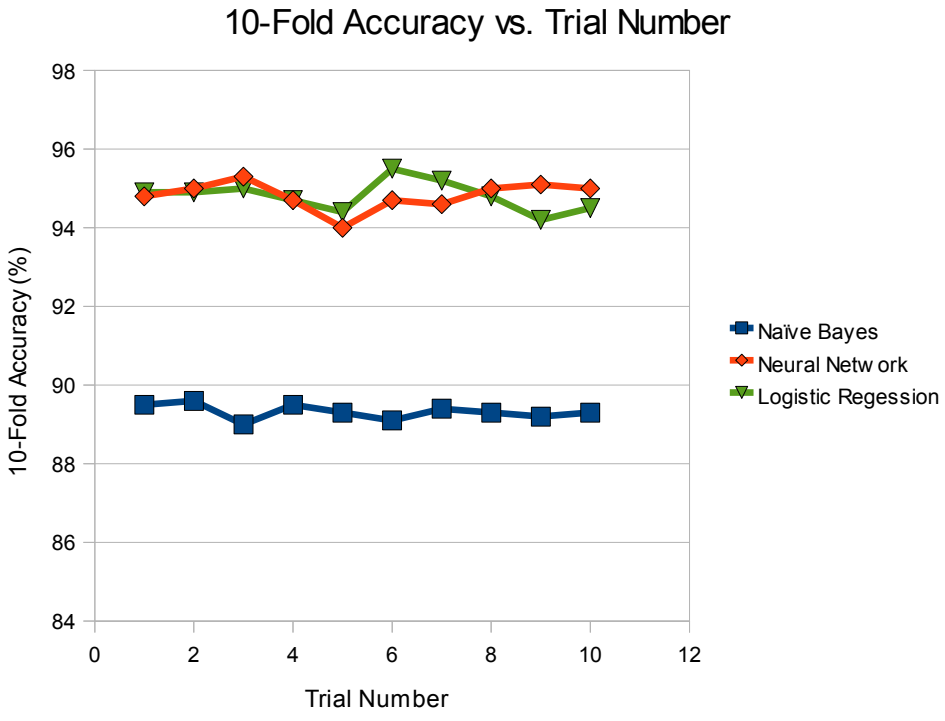

Figure 1. Per-trial accuracy using 10-fold cross validation for each system over the data set.

Table 1 shows the mean and standard deviation of the accuracy taken from the above graph. As can be seen, for this experiment the multilayer neural network and logistic regression approaches perform at almost the same level in terms of overall accuracy, but the latter classifier is slightly more variant in terms of its results. More notably, we see that the naïve Bayes classifier, though most able to consistently label the instances at a given level of accuracy, turned out to be less accurate over the ten trials as compared to the other two classifiers.

	Naïve Bayes	**Neural Network**	**Logistic Regression**
Mean	89.32%	94.82%	94.81%
Standard Deviation	0.187%	0.358%	0.384%

Table 1. Mean and standard deviation for overall accuracy for each classifier.

As another means of comparison, we provide the following receiver operating characteristic (ROC) curves for each of the above classifiers. Notice that these results parallel the accuracy plot from before, assuming equal misclassification costs for positive and negative instances. The full set of ROC curves is given in Figure 2 while a zoomed-in section of the plot is given in Figure 3.

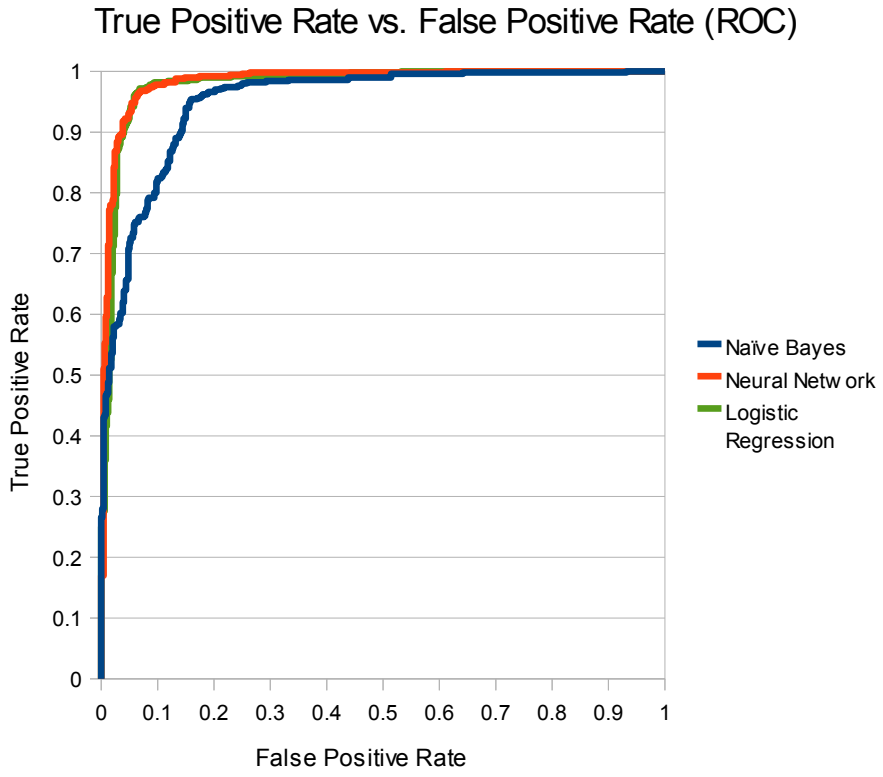

Figure 2. ROC curves showing true and false positive rates for each classifier.

Figure 3, which shows the upper-left quarter of the plot from Figure 2, is perhaps the

more informative graphic. Here it can be seen that the multilayer neural network and logistic regression approaches are fairly comparable in terms of accuracy-related metrics. Interestingly, it would seem that if we imposed differential misclassification costs such that, say, a false positive was less desirable than a false negative, then in either direction the multilayer neural network approach may perform marginally better than the logistic regressor given these curves.

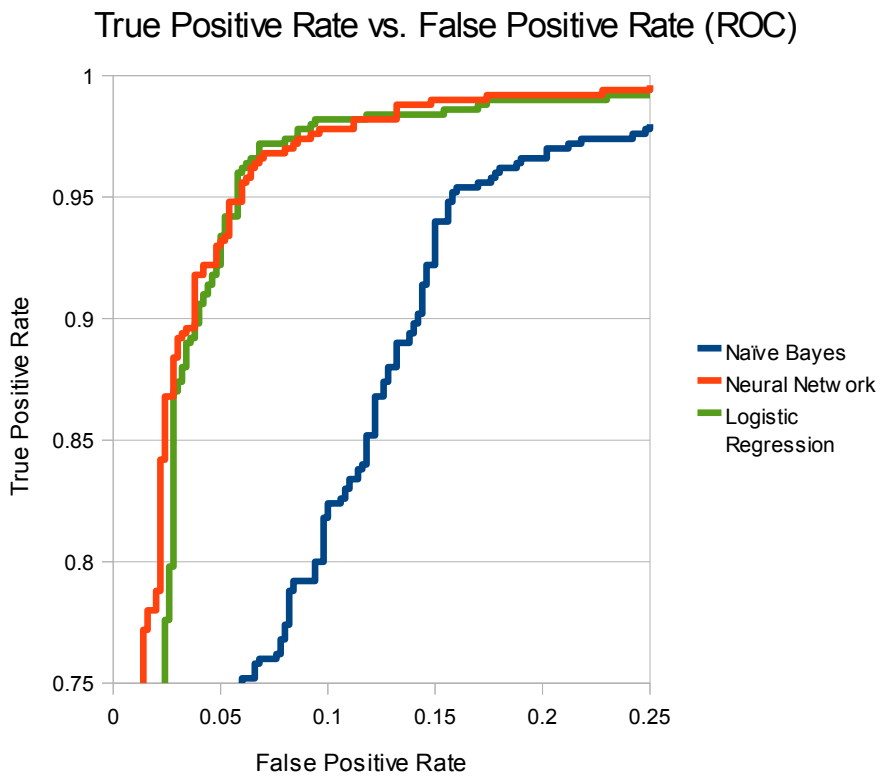

Figure 3. A zoomed-in section of the previous ROC curve.

The previous experiment sought to produce a comparison between the classifiers on a spam data set for which the instances were more or less evenly distributed in terms of message timestamp. In reality, ham and spam messages are delivered consistently over time, but the contents of spam messages in particular can vary considerably over different time frames. This is due in part to the current widespread use of adaptive spam filters, as spammers attempt to change the contents of their messages in order to circumvent such automated mechanisms. As such, we devised another experiment in which classifiers were trained over a portion of the data set for which the timestamps of the messages were from earlier in the year and subsequently tested over instances from later in the year. This was done to determine how well each classifier is able to account for adaptations in spam content over time.

For this second experiment, we divided the original data set in half, separating instances by timestamp and broadly placing them into 'early' and 'late' divisions. From the 'early' division, we took the 400 earliest instances, evenly split between ham and spam, and thereby formed the training set for this experiment. The test set was created by taking the latest 100 instances from the 'late' division and ensuring that the ham to spam ratio was again even within the set. This particular 80/20 split was chosen in order to ensure that sufficient data would be used during

training while maintaining a wide enough separation of timestamps to guarantee that the messages from the test set differed in content to a satisfactory degree as compared to the messages in the training set.

Using this training and test set, we evaluated each classifier's performance as before, with the learning rate, number of training epochs, and number of discrete bins selected empirically for optimal accuracy while maintaining a one minute running time restriction. Figure 4 gives the test set accuracy for each classifier over ten trials, and aggregate statistics are given in Table 2.

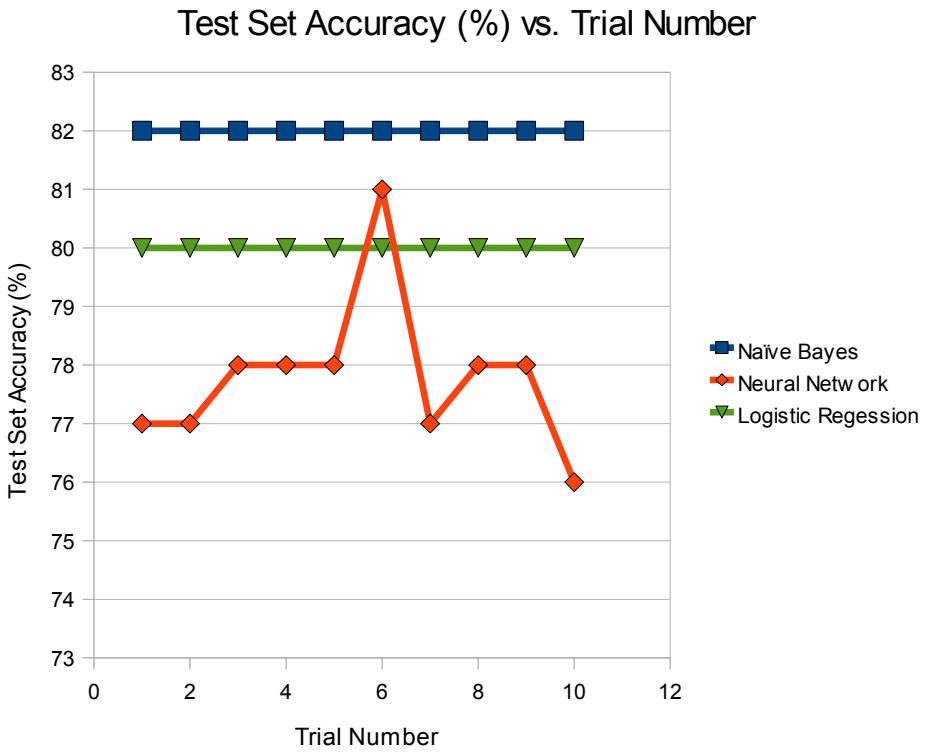

Figure 4. Accuracy over the 'late' test set for each classifier trained on the 'early' training set over ten trials.

	Naïve Bayes	Neural Network	Logistic Regression
Mean	82.0%	77.8%	80.0%
Standard Deviation	0.0%	1.32%	0.0%

Table 2. Mean and standard deviation for overall 'late' test set accuracy for each classifier.

This plot shows an appreciable decrease in accuracy as compared to the first experiment, which is consistent with what we would expect as there should be a detectable difference between the spam in the training set versus the test set given the time gap that we had imposed between the two sets. Interestingly, the naïve Bayes classifier appears to be consistently more accurate than its discriminative counterparts, which is in line with our original hypothesis. Both the naïve Bayes and logistic regression classifiers gave the same results across all trials while the

multilayer neural network had more variation in its output. This is likely due to the high density of random weights involved in the initialization of the network at the start of each trial.

4. Discussion

The results from these experiments offered some interesting contrast to our initial hypotheses. From the first experiment, it would seem that both discriminative approaches, the multilayer neural network and the logistical regressor, outperformed the naïve Bayes method when considering spam classification over a set of instances without taking into account timestamp order. In retrospect, it makes sense for the logistic regressor to have done better in this experiment compared to the naïve Bayes classifier given that this was a larger training set; given valid modeling assumptions on the part of the naïve Bayes approach, both classifiers should be equally accurate given enough training examples since they are of the same functional form. Since the features for this data set rely on the content of written messages, it is likely that the conditional independence assumptions that naïve Bayes relies on (i.e., its modeling assumptions) were not valid, resulting in the difference in performance between the generative and the discriminative classifiers that we see here.

While the first experiment generally invalidated our first hypothesis' prediction that the naïve Bayes classifier would have the best performance, it did offer some support for the rationale behind our second hypothesis. Although the multilayer neural network did not outperform the logistic regressor in this experiment as we had anticipated, our reasoning that the neural network, a non-linear classifier, would do at least as well at handling linearly separable instances as the logistic regressor, a linear classifier, appears to be sound. It should be noted, however, that owing to the neural network's additional topological complexity coupled with the generally longer running time of the backpropagation algorithm, it performs much more slowly than the linear regression classifier over a fixed number of training epochs.

The second experiment seemed to lend credence to our first hypothesis, though in a more specific context than we had anticipated. Compared to the two discriminative approaches, naïve Bayes performed better and was generally more consistent in its results when accounting for separation of data by timestamp. However, given the relatively narrow difference between the results for the naïve Bayes classifier and those for the multilayer neural network and logistic regressor, it is difficult to discern whether or not this is significant. It could be the case that the naïve Bayes paradigm is able to model the feature space in a way that is more flexible than a discriminative model and thus responds better to the spam adaptations experienced here, or perhaps it is simply more resistant to overfitting based on the training data as compared to the other approaches. Without additional data toward this end, it is difficult to tell for certain.

With regards to our experimental procedure, probably the aspect which prevented us from generating more confident results was a lack of diverse data. Although we were able to garner a thousand instances from the author's workplace inbox, in order to make better comparisons between time spans as we did in the second experiment, we would need more messages fitting into the 'early' and 'late' divisions to produce more robust results. As well, since much of our parameter setting for each of the classifiers was based on manual tuning, it would be much better from an experimental standpoint to have been able to find the optimal settings computationally, although for the purposes of this project such a thing would likely be out of scope.

From the results of this project, we see a number of interesting possibilities in terms of improvements and expansions to the original design. First and foremost would be to investigate

the adaptive capabilities of each classifier given spam over time, as we did in the second experiment, but with a larger sample of data and with a more diverse selection of time slices; for example, instead of simply separating instances by their timestamps being 'early' or 'late', one could do a month-by-month comparison. Another possibility might be to take into consideration the importance of preventing false positives versus false negatives in a real-world application; in practice, the misclassification cost of having a spam message end up in one's inbox is much smaller than that of having a legitimate message delivered to one's junk folder.

More broadly, it might be worthwhile to look into more sophisticated means of engineering features and see if, for example, providing an encoding of features that moves the decision boundary to a non-linear space would provide for a more robust representation of the data set. Toward this end, it would be straightforward to use this new feature set to evaluate multilayer neural networks of increasing complexity, with the expectation that one could outperform the classifiers from this project due to its ability to separate instances more accurately based on this advanced feature set. Since the multilayer neural network used in this project had only one hidden layer, it would also be interesting to see how different topologies for these proposed networks would fare at the task. Of course, this depends largely on the form of the features being considered during classification.

5. Conclusion

For this project, we designed and implemented three classifiers, one each for the naïve Bayes, logistic regression, and multilayer neural network approaches, and evaluated each on a spam classification task. In our first experiment, we looked at how each classifier performed over a data set of a thousand instances using 10-fold cross validation over ten trials. We found that both the logistic regressor and the multilayer neural network, both discriminative learners, did consistently better than the naïve Bayes classifier, which went against our initial expectations. The second experiment carried out for this project involved comparing each classifier with regards to how well they were able to classify newer ham and spam messages after having trained on older messages. The results showed a marginal advantage to the naïve Bayes classifier as compared to the other two classifiers. This project can be seen as a preliminary step toward further investigations into the relative efficacy of spam classifiers from these different paradigms, as such aspects of this project as the feature set and the manner of dividing instances by time are good candidates for expanded study.

6. References

[1] Phys.org, "107 trillion emails sent last year: Pingdom." *Phys.org*, 2011. http://phys.org/news/2011-01-trillion-emails-year-pingdom.html.

[2] Apache Software Foundation, "The Apache SpamAssassin Project." *Apache Software Foundation*, 2011. http://spamassassin.apache.org/

[3] A. Ng and M. Jordan, "On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes." *NIPS*, 14 (2001), p. 605–610.

[4] D. Rumelhart, G. Hinton, and R. Williams. "Learning Internal Representations by Error Propagation." *Parallel Distributed Processing*, 1 (1987), p. 318-362.

[5] N. Friedman, D. Geiger, and M. Goldszmidt. "Bayesian Network Classifiers." *Machine Learning*, 29 (1997), p. 131-163.

[6] T. Mitchell. "Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression," in *Machine Learning*, 2nd ed. McGraw Hill, 2010. Available: www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf.
[7] StackExchange, "How to choose the number of hidden layers and nodes in a feedforward neural network?" StackExchange, 2010. http://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw
[8] A. Blum. *Neural Networks in C++*. NY: Wiley, 1992.
[9] M. Hopkins, et al. "Spambase Data Set." *UCI Machine Learning Repository*, Irvine, CA: University of California, School of Information and Computer Science, 1999. Available: http://archive.ics.uci.edu/ml/datasets/Spambase.