3

Content-based spam email classification using machine-learning algorithms

Eric P. Jiang

3.1 Introduction

With the rapid growth of the Internet and advances in computer technology, email has become a preferred form of communication and information exchange for both business and personal purposes. It is fast and convenient. In recent years, however, the effectiveness and confidence in email have been diminished quite noticeably by spam email, or bulk unsolicited and unwanted email messages. Spam email has been a painful annoyance for email users with an overwhelming amount of unwelcome messages flowing into their inboxes. Now, it has also evolved into a primary medium for spreading phishing scams and malicious viruses. The cost of spam in the United States alone in terms of decreased productivity and increased technical expenses for businesses has reached tens of billions of dollars annually.\footnote{http://www.spamlaws.com/spam-stats.html} Worldwide spam volume has increased significantly and during the first quarter of 2008, spam email accounted for more than nine out of every ten email messages sent over the Internet.\footnote{http://www.net-security.org/}
Over the years, various spam filtering technology and anti-spam software products have been developed and deployed. Some of them are designed to detect and stop spam email at the TCP/IP or SMTP level and may rely on DNS blacklists of domain names that are known to originate spam. This approach has been commonly used. However, it can be insufficient due to the lack of accuracy of the name lists, since spammers can now register hundreds of free webmail services such as Hotmail and Gmail and then rotate them every few minutes during a spam campaign. The other major type of spam filtering technology functions at the client level. Once an email message is downloaded, its content can be examined to determine whether the message is spam or not. Several supervised machine-learning algorithms have been used in client-side spam detection and filtering. Among them, naive Bayes (Mitchell 1997; Sahami et al. 1998), boosting algorithms such as logitBoost (Androulidakis et al. 2004; Friedman et al. 2000), support vector machines (SVMs) (Christianini and Shawe-Taylor 2000; Drucker et al. 1999a), instance-based algorithms such as k-nearest neighbor (Aha and Albert 1991), and Rocchio’s classifier (Rocchio 1997) are commonly cited. More recently, a number of other interesting algorithms for spam filtering have been developed. One uses an augmented latent semantic indexing (LSI) space model (Jiang 2006) and another applies a radial basis function (RBF) neural network (Jiang 2007).

This chapter considers five supervised machine-learning algorithms for an evaluation study of spam filtering application. The algorithms considered in this study include widely used ones with good classification results and some recently proposed methods. More specifically, we evaluate these five classification algorithms: naive Bayes classifier (NB), support vector machines (SVMs), logitBoost algorithm (LB), augmented latent semantic indexing space model (LSI) and radial basis function (RBF) networks.

Spam filtering is a cost-sensitive classification task since misclassifying legitimate email (a false positive error) is generally more costly than misclassifying spam email (a false negative error). Fairly recently, there have been several studies (Androulidakis et al. 2004; Zhang et al. 2004) surveying machine-learning techniques in spam filtering. Using a constant $\lambda$ to measure the higher cost of false positives, these studies have evaluated several algorithms on spam filtering by integrating the $\lambda$ value or a function of $\lambda$ into the algorithms through a variety of cost-sensitive adjustment strategies. This was done by increasing algorithm thresholds on spam confidence scores, adding more weights on legitimate training samples, or empirically adjusting algorithm decision thresholds using cross-validation. Different adjustment strategies have also been applied to different algorithms in the studies. Since all the algorithms were designed with cost-insensitive tasks in mind, applying such simple cost-sensitive adjustments on the algorithms can produce unreliable results. Apparently this insufficiency has been recognized and, for some algorithms, the studies reported only the best results among several adjustment trials.

This chapter provides a related study of five machine-learning algorithms on spam filtering from a different perspective. The main objective of the study is to learn whether and to what extent the algorithms are adaptable and applicable to the cost-sensitive email classification problem and to identify the characteristics of the algorithms most suitable for adaptability. In this study, we selected two benchmark email testing corpora for experiments that were constructed from two different languages and have reverse ratios of the number of spam emails to the number of legitimate emails in the training data. We also vary feature size in the experiments to analyze the usefulness of feature selection for these algorithms.

The rest of the chapter is organized as follows. In Section 3.2, the five machine-learning algorithms that are investigated for spam filtering applications are briefly described. In Section 3.3, several data preprocessing procedures, including feature selection and message representation, are discussed. Spam filtering is a cost-sensitive classification task and a related discussion of effectiveness measures is included in Section 3.4. We then compare the algorithms, using two popular email testing corpora. The experimental results and analysis are reported in Section 3.5, and an empirical comparison of the characteristics of the five classifiers is presented in Section 3.6. Finally, some concluding remarks are provided in Section 3.7.

### 3.2 Machine-learning algorithms

Spam email filtering is an application of automated text classification with two categories. A number of machine-learning algorithms, which have been successfully used in text classification (Sebastiani 2002), can also be applied in spam filtering. Given a collection of labeled email samples, these algorithms can learn from the samples to classify previously unseen email into the categories based on their content. The algorithms of NB, LB, SVM, augmented LSI, and RBF are among those that have achieved good performance for spam filtering. They are included in this study and are briefly described in this section.

In this chapter, we use $D = \{d_1, d_2, \ldots, d_n\}$ to denote a training set of email samples with size $n$ and $C = \{c_l, c_r\}$ the email categories ($c_l$ legitimate; $c_r$ spam). We assume each email message $d_i$ can be expressed as a numeric vector representing the weights of terms or features $d_i = (t_1, t_2, \ldots, t_m) \in \mathbb{R}^m$ (see Section 3.3.2).

#### 3.2.1 Naive Bayes

The NB classifier is a probabilistic learning algorithm that derives from Bayesian decision theory (Mitchell 1997). The probability of a message $d$ being in class $c$, $P(c|d)$, is computed as

$$P(c|d) \propto P(c) \prod_{k=1}^{m} P(t_k|c),$$

where $P(t_k|c)$ is the conditional probability of feature $t_k$ occurring in a message of class $c$ and $P(c)$ is the prior probability of a message occurring in class $c$. This chapter provides a related study of five machine-learning algorithms on spam filtering from a different perspective. The main objective of the study is to learn whether and to what extent the algorithms are adaptable and applicable to the cost-sensitive email classification problem and to identify the characteristics of the algorithms most suitable for adaptability. In this study, we selected two benchmark email testing corpora for experiments that were constructed from two different languages and have reverse ratios of the number of spam emails to the number of legitimate emails in the training data. We also vary feature size in the experiments to analyze the usefulness of feature selection for these algorithms. The rest of the chapter is organized as follows. In Section 3.2, the five machine-learning algorithms that are investigated for spam filtering applications are briefly described. In Section 3.3, several data preprocessing procedures, including feature selection and message representation, are discussed. Spam filtering is a cost-sensitive classification task and a related discussion of effectiveness measures is included in Section 3.4. We then compare the algorithms, using two popular email testing corpora. The experimental results and analysis are reported in Section 3.5, and an empirical comparison of the characteristics of the five classifiers is presented in Section 3.6. Finally, some concluding remarks are provided in Section 3.7.
is estimated by applying a sigmoid function, which is also known as the logit transformation, to the response of the ensemble that has been built so far, i.e.

\[
P(c|d) = \frac{e^{F(d)}}{1 + e^{F(d)}}, \quad F(d) = \frac{1}{2} \sum_{m} f_m(d).
\]

Once the iteration terminates and the final ensemble \( F \) is created, the classification of target email messages is determined by the probability in Equation (3.4).

A popular base learner choice for LB is decision stump, a one-level decision tree that uses an attribute in training data to classify training samples into categories. In text classification, since we deal with continuous attributes, the decision tree is actually a threshold function on one of the data attributes and hence it becomes a regression stump (Andoutopoulos et al. 2004). It can be shown that the LB algorithm maximizes the probability of the data with respect to the ensemble if each base learner \( f_m \) is determined by minimizing the squared error on the fitted regression of weighted training data (Witten and Frank 2005). The model's iteration number \( m \) is specified by the user and we set it to 50, which is the smallest feature size used in this study.

### 3.2.3 Support vector machines

SVMs (Christianini and Shawe-Taylor 2000) have been considered the most promising algorithm in text classification. The algorithm uses linear models to implement nonlinear category boundaries by transforming a given instance space into a linearly separable one through nonlinear mappings. In the transformed space, an SVM constructs a separating hyperplane that maximizes the distance between the training samples of two categories. This is done by selecting two parallel hyperplanes that are each tangent to at least one sample of its category; such samples on the tangential hyperplanes are called the support vectors. The distance between the two tangential planes is the margin of the classifier, which is to be maximized, and that is why a linear SVM is also known as a maximal margin classifier.

Assume the class variable for the \( i \)th training sample is \( c_i = \{1, -1\} \), indicating the spam (1) or legitimate (-1) category, respectively. A hyperplane in the sample space can be written as

\[
w \cdot d + b = 0,
\]

where \( w \) is a normal vector that is perpendicular to the hyperplane, and \( b \) is a bias term. If the given training data is linearly separable, we can select two hyperplanes that contain no points between them and then maximize the distance (margin) between the hyperplanes, which is \( 2/\|w\| \). Maximizing the margin is equivalent to solving the following constrained minimization problem:

\[
\min_{w} \frac{\|w\|^2}{2}, \quad \text{subject to } c_i(w \cdot d_i + b) \geq 1.
\]
The optimization problem in Equation (3.6) can be solved by the standard Lagrange multiplier method with the new objective function:

$$\frac{\|w\|^2}{2} - \sum_{i} \lambda_i [c_i (w \cdot d_i + b) - 1].$$  \hspace{1cm} (3.7)

Since the Lagrangian involves a large number of parameters, this is still a difficult problem. Fortunately, the problem can be simplified by transforming the Lagrangian in Equation (3.7) into the following dual formation that contains only Lagrange multipliers:

$$\max \sum \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j c_i c_j d_i \cdot d_j, \text{ subject to } \lambda_i \geq 0, \text{ and } \sum_{i} \lambda_i c_i = 0.$$  \hspace{1cm} (3.8)

The dual optimization problem can usually be solved by using some numerical quadratic programming techniques such as the sequential minimal optimization algorithm (Platt 1999). The terms $\lambda_i$ from Equation (3.8) are used to define the decision boundary

$$\left( \sum_{i} \lambda_i c_i d_i \cdot d \right) + b = 0.$$  \hspace{1cm} (3.9)

In order to deal with the cases where the training samples cannot be fully separated and also small misclassification errors are permitted, the so-called soft margin method was developed for choosing a hyperplane that intends to reduce the number of errors committed by the decision boundary while maximizing the width of the margin. The method introduces a positive-valued slack variable $\xi$ that measures the degree of misclassification error on a sample and solves the following modified optimization problem:

$$\min_w \frac{\|w\|^2}{2} + C \sum_{i} \xi_i, \text{ subject to } c_i (w \cdot d_i + b) \geq 1 - \xi_i.$$  \hspace{1cm} (3.10)

where a linear penalty function is used and $C$ is a user-specified constant that determines an error tolerance level. In our experiments, we set $C = 1$.

The linear SVM described above can be extended into a nonlinear classifier. Conceptually, we could just transform the training data (where no linear decision boundaries can be found) to a new feature space so that a linear decision boundary can be constructed to separate the data in the transformed space. However, this feature transformation approach raises a few issues about high feature dimensionality and high computational requirements. Alternatively, nonlinear classifiers can be created by applying a procedure similar to the linear ones to construct maximum margin hyperplanes, except that every dot product in the transformed space is replaced by a kernel function in the original feature space. Computing the dot products using kernels is considerably cheaper than using the transformed features. Several different kernel functions have been proposed and, for text classification, it seems that the SVM with a simple linear kernel performs comparably to nonlinear alternatives (Joachims 1998). An SVM with a linear kernel is used in our evaluation.

### 3.2.4 Augmented latent semantic indexing spaces

Latent semantic indexing (LSI) (Deerwester et al. 1990) is a well-known information retrieval technique. By deploying a rank-reduced feature–document space through the singular value decomposition (SVD) (Golub and van Loan 1996), it effectively transforms individual documents into their semantic content vectors to estimate the major associative patterns of features and documents and to diminish the obscuring noise in feature usage (Berry et al. 1995).

LSI can be used as a learning algorithm for spam filtering by replacing the notion of query relevance with the notion of category membership. An experiment of this approach on the Ling-Spam corpus was reported in Gee (2003) and it constructs a single LSI space to accommodate both spam and legitimate email training data. This simple application has some drawbacks (Jiang 2006). LSI itself is a completely unsupervised learning algorithm and when it is applied to (supervised) spam filtering, valuable category discriminative information embedded in training data should be extracted and integrated in model learning to boost classification accuracy. There are several approaches that can be used toward this goal. For instance, we can select distinctive features by exploring their category distributions (see Section 3.3) and introduce two separate LSI learning spaces (one for each email category). Feature selection also helps reduce computational requirements due to the SVD algorithm in the model.

For a given email training set, each of the two rank-reduced spaces can be constructed by using the data of its respective category and conceptually it would provide a more accurate category content profile than that produced from a single combined space. In practice, however, this dual-space approach may still encounter difficulties in classifying some email messages since many spam messages are purposely crafted to look legitimate and to mislead spam filters. This has been verified by our extensive experiments. In order to ameliorate this problem, a new model that uses augmented LSI learning spaces was proposed in Jiang (2006). More precisely, for each constructed category LSI space, this model augments the space with a small number of the training samples that are closest to the category in appearance but actually belong to the other category in label. This augmented LSI space model can effectively help classify those difficult target messages correctly, which are similar to the augmented samples used in the training, while maintaining accurate classification of other messages.
Expansion of the augmented training samples is carried out by cluster centroids. For each email category, we construct one or multiple clusters. For each cluster \( c_\beta \), its centroid is computed as

\[
a_{\beta} = \frac{1}{k} \sum_{i=1}^{k} d_{\beta i}, \quad d_{\beta i} \in c_\beta,
\]

and it can be used to represent the most important topic covered in the cluster (Jiang 2006). Once the cluster centroids of a category \( c \) are identified, all training samples from the other category are compared against the centroids and the most similar ones are then chosen to add to the training set of \( c \). Selecting the sizes of clusters and augmented samples of a category can vary depending on the data to be learned. The cluster size can also be set by a silhouette plot (Kaufman and Rousseeuw 1990) on a given training dataset. In our experiments, we used the augmented sample sizes of 18 and 70 for the corpora PU1 and ZH1 (see Section 3.5), respectively.

To use two separate augmented LSI spaces for classification, several approaches have been considered and evaluated in Jiang (2006) that coordinate and classify target email messages into their respective classes. For a given target message, the first approach simply projects it onto both LSI spaces and then uses the most semantically similar training sample to decide the class for the message. The second approach classifies the message similarly but by applying a fixed number of the top most similar training samples in the spaces and using either the sum or average of computed similarity values from both classes to make its classification decision. The third approach is a hybrid one that intends to combine the ideas of the first two methods and also to mollify some of their shortcomings. Essentially, it determines the class for the target message by linearly balancing the votes or decisions made by the first two methods. Experiments indicate that in general the hybrid approach delivers significantly better classification results (Jiang 2006) and it is used in the study.

### 3.2.5 Radial basis function networks

RBF networks have many applications in science and engineering and can also be used to build learning models for filtering spam email (Jiang 2007). A typical RBF network has a feedforward connected structure of three layers: an input layer, a hidden layer of nonlinear processing neurons, and an output layer (Bishop 1995). For email classification, the input layer of the network has \( n \) neurons and it takes input training samples \( d \). The hidden layer contains \( k \) computational neurons; each neuron can be mathematically described by an RBF \( \phi_i \) that maps a distance between two vectors in the Euclidean norm into a real value:

\[
\phi_i(x) = \phi(||x - a_i||^2), \quad i = 1, 2, \ldots, k,
\]

where \( a_i \) are the RBF centers in the input sample space and, in general, \( k \) is less than the size of training samples. The output layer of the network has two neurons that produces the target message category according to

\[
c_j = \sum_{i=1}^{k} w_{ij} \phi_i(x), \quad j = 1, 2,
\]

where \( w_{ij} \) is the weight connecting the \( i \)th neuron in the hidden layer to the \( j \)th neuron in the output layer. The neuron activation \( \phi_i \) is a nonlinear function of the distance; the closer the distance, the stronger the activation. The most commonly used basis function is the Gaussian

\[
\phi(x) = e^{-\frac{x^2}{2\sigma^2}},
\]

where \( \sigma \) is a width parameter that controls smoothness properties of the basis function.

In the spam filtering model (Jiang 2007), the network parameters, i.e., centers, widths, and weights, are set by a two-stage training procedure, which is computationally efficient. The first stage of training is to form a representation of the density distribution in input space in terms of the parameters of the RBFs. The centers \( a_i \) and widths \( \sigma \) are determined by relatively fast and unsupervised clustering algorithms, clustering each email category independently to obtain \( k \) basis functions for the category. In general, the larger the value of \( k \), the better the classification outcomes and, of course, the higher the cost it carries in network training. With the computed and fixed centers and widths for the hidden layer, the second stage of training selects the weights of the output layer by a logistic regression procedure. Once all network parameters are determined, the model can be deployed to target email messages for classification, and classification outcomes from the network are computed by a weighted sum of the hidden layer activations, as is shown in Equation (3.13).

Recently, an RBF-based semi-supervised text classifier has also been developed (Jiang 2009). It integrates a clustering-based expectation maximization algorithm into the RBF training process and can learn for classification from a very small number of labeled training samples and a large pool of additional unlabeled data effectively.

### 3.3 Data preprocessing

In this section, we begin with some data preprocessing procedures that include feature selection and message representation, followed by a discussion of classification effectiveness measures for spam filtering.

#### 3.3.1 Feature selection

As in general text classification, appropriate feature selection can be quite useful in aiding email classification. A term or feature is referred to as a word, a

number, or a symbol in an email message. In spam filtering, features from training samples are selected according to their contributions to profiling legitimate or spam messages and those unselected features are removed from the data for model learning and deployment. The objectives of feature selection are twofold. On one hand, it is designed for dimensionality reduction in the message feature space. Dimensionality reduction aims to trim down the number of features to be modeled while the content of individual messages is still preserved. It generally helps speed up a model training process. On the other hand, feature selection intends to filter out irrelevant features, helping build an accurate and effective model for spam filtering. This is particularly valuable to certain machine-learning algorithms such as RBF networks, which treat every data feature equally in their distance computations and therefore are somewhat incapable of distinguishing relevant features from irrelevant ones.

Two steps of feature selection are used in our experiments. First, for a given set of training data, features are extracted and selected with an unsupervised setting. This is carried out by removing the stop or common words and applying a word stemming procedure. Then, the features with low message frequencies or low corpus frequencies are eliminated from the training data, as these features may not help much in differentiating messages for categories and may add some obscuring noise in email classification. The selection process also removes those features with very high corpus frequencies in the training data as many of these features distribute almost equally between spam and legitimate categories and may not be valuable in characterizing the email categories. Next, features are selected by their frequency distributions between spam and legitimated training messages. This supervised feature selection procedure intends, using those labeled training samples, to further identify the features that distribute most differently between the categories.

There are several supervised feature selection methods that have been widely used in text classification (Sebastiani 2002). They include the chi-square statistic (CHI), information gain (IG), and odds ratio (OR) criteria. The IG criterion quantifies the amount of information gained for category prediction by knowledge of the presence or absence of a feature in a message. More precisely, IG of a feature \( t \) about a category \( c \) can be expressed as

\[
IG(t, c) = \sum_{c'} \sum_{t' \in \{c, \bar{c}\}} P(t', c') \log \frac{P(t', c')}{P(t') P(c')},
\]

where \( P(c') \) and \( P(t') \) denote the probability that a message belongs to category \( c' \) and the probability that a feature \( t' \) occurs in a message, respectively, and \( P(t', c') \) is the joint probability of \( t' \) and \( c' \). All probabilities can be estimated by frequency counts from the training data. Another popular feature selection method is CHI. It measures the lack of independence between the occurrence of feature \( t \) and the occurrence of class \( c \). In other words, features are ranked with respect to the quantity

\[ CHI(t, c) = \frac{n[P(t, c)P(\bar{c}) - P(t, \bar{c})P(\bar{t}, \bar{c})]^2}{P(t)P(\bar{t})P(c)P(\bar{c})}, \]

where \( n \) is the size of training data \( D \) (see Section 3.2) and the probability notations have the same interpretations as in Equation (3.15). For instance, \( P(\bar{c}) \) represents the probability that a message does not belong to category \( c \). The third feature selection criterion, OR, has also been used in text classification and it measures the ratio of the odds of term \( t \) occurring in a message of class \( c \) to the odds of the term not occurring in \( c \) and can be defined as

\[ OR(t, c) = \frac{P(t|c)(1 - P(\bar{t}|\bar{c}))}{(1 - P(t|c))P(\bar{t}|\bar{c})}. \]

The effectiveness of the feature selection methods for text classification has been studied and compared, e.g. by Yang and Pedersen (1997), and some experiments with the criteria described above have also been conducted in this study. Among these three feature selection methods, our experiments suggest that the IG measure produces more stable classification results, so we used it in the selection process.

Through feature selection, the feature dimensionality of a training dataset can be reduced significantly. For instance, in the experiments with PUI (see Section 3.5.1) the original feature size of the corpus, which is over 20000, can be trimmed down to tens, hundreds, and thousands.

### 3.3.2 Message representation

After feature selection, each message is encoded as a numeric vector whose elements are the values of the retained feature set. Each feature value is associated with a local and global feature weight, representing the relative importance of the feature in the message and the overall importance of the feature in the corpus, respectively. Our experiments indicate that feature frequencies are more informative than a simple binary coding (which, for instance, is used in Zhang et al. (2004)) in the context of email classification.

There are several choices to weight a feature or term locally and globally based on its frequencies. For a given term \( t \) and document \( d \), the traditional 'log(tf)−idf' term weight is defined as

\[ w_{t,d} = \log(1 + tf_{t,d}) \log \frac{|D|}{df_t}, \]

where \( tf_{t,d} \) is the term frequency (tf) of \( t \) in \( d \), \( df_t \) is the document frequency (df) of \( t \), or the number of documents in a collection \( D \) that contain \( t \), and \(|D|\) is the size of the collection. The second component on the right hand side
of Equation (3.18) represents the inverse document frequency (idf) of \( t \). This term weighting scheme is used in this work and it produces good classification results.

### 3.4 Evaluation of email classification

The effectiveness of a text classifier can be evaluated in terms of its precision \( (p) \) and recall \( (r) \) measures. For a classifier and with respect to a category \( c \), if the numbers of true positive, false positive, and false negative decisions on category \( c \) from the classifier are \( tp, fp, \) and \( fn \), respectively, then the precision and recall are defined as

\[
 p = \frac{tp}{tp + fp}, \quad r = \frac{tp}{tp + fn}.
\]  

(3.19)

In brief, the precision measure is gauged by the percentage of documents classified to \( c \) which actually are, whereas the recall is quantified by the percentage of documents from \( c \) that are categorized by the classifier. Clearly, these two quantities trade off against one another and one single measure that balances both is the \( F \)-measure, which is the weighted harmonic mean of precision and recall. With an equal weight for both precision and recall, we have the commonly used \( F_1 \) measure

\[
 F_1 = \frac{2pr}{p + r}.
\]  

(3.20)

All these effectiveness measures, however, do not take a possible unbalanced misclassification cost into consideration. Spam email filtering can be a cost-sensitive learning process in the sense that misclassifying a legitimate message to spam (false positive) is typically a more severe error than misclassifying a spam message to legitimate (false negative). In reality, if a legitimate message is mistakenly classified and placed into a user’s trash-mail box, then the user may not find this out for a short or long period of time and, depending on how important the message is, a delayed reading of the message could come with some negative consequences. In our experiments, an accuracy measure that uses a weight \( \lambda \) to reflect the unbalanced cost between false positive and false negative errors, or the weighted accuracy (Androutsopoulos et al. 2004), is used as the effectiveness criterion and it can be defined as

\[
 WA(\lambda) = \frac{\lambda \cdot tp + m}{\lambda \cdot (m + fp) + (tp + fn)},
\]

(3.21)

where the quantities \( tp, fp, \) and \( fn \) are the same as in Equation (3.19), \( m \) denotes the true negative classification count, and \( \lambda \) is a cost parameter. The \( WA \) formula assumes that a false positive error is \( \lambda \) times more costly than a false negative one. We use \( \lambda = 1 \) for the case where both false positive and false negative errors have an equal cost and also a value of \( \lambda \) that is greater than one, such as \( \lambda = 9 \), to indicate a higher cost of false positive errors. It is still arguable if such a higher cost in spam filtering can be quantified by a simple constant (Hidalgo 2002), and the cost should perhaps depend on several variable external factors. In this study, we use \( \lambda = 9 \) (or any other number in a similar quantity) just as a value to illustrate whether or not and how the effectiveness of the algorithms may change when a cost-sensitive condition is imposed.

### 3.5 Experiments

In this section, we use two benchmark email testing corpora to compare the efficacy of the five machine-learning algorithms, discussed in Section 3.2, for spam email filtering and provide the experimental results and analysis. Note that the input data to the classifiers is the preprocessed message vectors after both feature selection and feature weighting.

#### 3.5.1 Experiments with P1U

P1U is a benchmark spam testing corpus that contains a total of 1099 real email messages received by a single email user over a certain period of time (Androutsopoulos et al. 2004) and it is partitioned into 618 legitimate and 481 spam messages. The messages in the corpus have been preprocessed with all attachments, HTML tags, and header fields, except for subject lines which were removed, and the retained words in the email subject line and body text were encoded numerically for privacy protection.

There are a few other publicly accessible spam datasets such as the 2005 TREC spam corpus that can be used for spam filtering evaluation. However, most of them were aggregated from multiple different email sources or recipients, and some of the large ones were constructed by simply adding some newly gathered email messages to what had been collected. For very understandable privacy reasons, it has been a challenge for IT researchers to find coherent, reliable, and updated public email data, which can reflect what an average email user receives, for conducting experiments and producing meaningful and comparable testing results.

It should be pointed out that, in this study, we use only email subject line and body text as the email content. This is a constraint imposed by construction of the corpora we used in the experiments. The machine-learning algorithms investigated in this chapter, however, can plainly be applied to broader email content. As noted by several previous studies, e.g., Zhang et al. (2004), the features from other email text such as headers are indeed useful in discriminating spam email. Therefore, we expect that the classification accuracy of the algorithms presented in this section would be further increased if we were to use the broader content that includes email header fields.

The experiments on P1U are performed using 10-fold cross-validation. That is, the corpus is partitioned into 10 equally sized subsets and each experiment
takes one subset for testing and the remaining ones for training and the process repeats 10 times with each subset taking a turn for testing. The effectiveness is then evaluated by averaging over the 10 experiments, delivered as an average weighted accuracy defined in Equation (3.21). Various feature sizes are also used in the experiments that range from 50 to 1650 with an increment of 100.

Classification effectiveness of the five algorithms, measured by the average weighted accuracy over all feature sizes that have been considered, is shown in Figure 3.1 ($\lambda = 1$) and Figure 3.2 ($\lambda = 9$), respectively. The case of $\lambda = 1$ may reflect classification efficacy of the algorithms for general cost-insensitive learning with a small number of classes. Figure 3.1 shows that RBF performs very well over small feature sizes, but, along with LB, it produces less accurate classification than all other three classifiers at large feature sizes. On the other hand, LSI behaves in a fairly opposite way: it is the least accurate classifier over small feature sizes but achieves good accuracy at large feature sizes. The relatively stable performance of NB, SVM, and LB through all feature sets can be observed, where NB is the top performer, followed closely by SVM and then LB at a distance.

Now, we turn to the case of $\lambda = 9$ and we intend to use the generated weighted accuracy values to demonstrate whether or not and how the accuracy results of an algorithm change when a false positive error is to be punished more than a false negative error or a cost-sensitive condition is imposed. The changes, if any, should ultimately depend on how well the algorithm can profile legitimate messages and make small numbers of false positive errors. For both NB and LB classifiers, their accuracy values in this case are not significantly different from those in Figure 3.1 and, relatively, their false positive errors are comparable to their false negative ones. Similar observations can also be made for SVM. On the other hand, since LSI, followed very closely by RBF, carries somewhat smaller numbers of false positive errors than other classifiers, its accuracy values are lifted for it to become the top performer. A detailed analysis of LSI and RBF on their error counts suggests that a richer feature set generally helps the classifiers characterize legitimate messages and improve classification of the category. But it may not be useful for them to improve their classification of spam messages. One possible explanation for this phenomenon may be related to the vocabularies used in the respective email categories. It is hypothesized that spam email has a strong correspondence between a small set of features and the category, while legitimate email likely carries more sophisticated characteristics. The spam category could attain good classification with a small vocabulary while the legitimate category requires a large vocabulary, which can be assisted by feature expansion.

### 3.5.2 Experiments with ZH1

In this subsection, we present the experiments of the five classifiers on a Chinese spam corpus ZH1 (Zhang et al. 2004). The experiments aim to demonstrate the capability of individual classifiers to classifying email written in a language with a different linguistic structure. Chinese text does not have explicit word boundaries like English, and words in the text can be extracted by some specially designed word segmentation software (Zhang et al. 2004). The construction of corpus ZH1 is very similar to PUI where ZH1 is made up of 1205 spam and 428 legitimate email messages. All messages in the corpus are also numerically encoded. Note that, in contrast to PUI, ZH1 has more spam email than legitimate email in the corpus and this helps examine whether or not and how the classifiers are possibly influenced in their model learning by unbalanced training sample sizes.
between the categories. Experiments on ZH1 are also performed using 10-fold cross-validation and the same feature sets as those with PU1.

Figure 3.3 and Figure 3.4 show the average weighted accuracy values obtained by all five classifiers over the feature sizes for $\lambda = 1$ and $\lambda = 9$, respectively. For the case of equal misclassification cost ($\lambda = 1$), Figure 3.3 indicates that SVM and LB perform best over most feature sizes, followed by LSI and then RBF; in this case, NB evidently fails to be comparable. When a higher cost on false positive errors is considered ($\lambda = 9$), similar observations can be made from

Figure 3.3  Average weighted classification accuracy with $\lambda = 1$ (ZH1).

Figure 3.4  Average weighted classification accuracy with $\lambda = 9$ (ZH1).

3.6 Characteristics of classifiers

In comparison to general text classification, spam email filtering represents a special, cost-sensitive, and very challenging classification task. It has two categories to be classified. The cost of the two types of misclassification errors is different and many spam messages are purposely and carefully constructed to look very much like legitimate ones. Though both spam and legitimate email messages may have a similar appearance, there may be still some important and different characteristics for each email category that should not be overlooked. For instance, in contrast to spam email, legitimate email has in general a broader vocabulary and also perhaps more eclectic subject matter. Ideally, a successful machine-learning algorithm used in this particular classification domain should utilize potential differences between the email categories and, more importantly, should be capable of profiling legitimate messages accurately and carry only a small number of false positive misclassification errors.

As in many other applications of machine learning, declaring one algorithm as the best for spam filtering is a difficult task and perhaps almost impossible. The experiments and analysis conducted in this study, however, have revealed some interesting characteristics among the five classifiers investigated. They are summarized below.

Naive Bayes (NB). This classifier is simple and the fastest in model learning among the five classifiers. It can work well for text classification. Since the algorithm assumes that individual features are completely independent of one another, the classifier can benefit from effective feature selection, which is demonstrated in the PU1 experiments. In the same vein, NB can perform poorly if it is applied to a dataset where there are some observable dependencies among features. One possible explanation for the inadequate performance of NB on ZH1 is the language on which the corpus is based. Chinese is a language with a vast vocabulary and it is extremely difficult to automatically extract meaningful words or features correctly from a Chinese document; many Chinese words are also polysemous (the words can have very different meanings depending on the context in which they are used). All of these language characteristics may contribute to inaccurate probability estimation and heavy feature dependencies, which can inevitably reduce the power of the NB algorithm.

LogitBoost (LB). As a boosting algorithm, LB combines multiple simple base learners (decision stumps in this case) iteratively to make a powerful classifier. Although the base learner has a very simple structure, the ensemble construction can still be very time consuming. The success of LB on text classification or spam filtering seems to depend on the dataset but generally LB delivers competitive results. One interesting and unique characteristic of the method is its insensitivity
to feature size and large feature sizes may not help improve its classification accuracy. Hence, it seems that a relatively small feature size such as 250 could be used for the model training. Finally, the learning ability of the classifier for profiling a category appears to be influenced by the size of available training samples of the category.

SVM. As reported by several previous studies, SVM is a very stable classifier and is also scalable to feature dimensionality. In this study, SVM consistently performs as the best or as a very competitive classifier, in particular when cost-sensitive classification is considered. The linear SVM used in this study is also relatively fast in model training.

Augmented latent semantic indexing spaces (LSI). The LSI model constructs two separate rank-reduced and augmented learning spaces, one for each email category. In this study, the model has been demonstrated to be a very reliable classifier and it consistently delivers competitive classification results. The model also seems well suited to cost-sensitive spam filtering and this could be in part due to its integrated clustering component for constructing the augmented LSI spaces. Good performance of the classifier generally requires a feature size of about 500 or larger. Algorithm training can be expensive if the feature size becomes very large.

Radial basis function networks. The RBF-based classifier performs reasonably well, especially when it is evaluated as a cost-sensitive learning algorithm. This is likely contributed by the clustering process used in its first stage of network training. The model’s performance appears to be affected by the clustering accuracy and, in addition, the classifier seems to be sensitive to feature size, so any excessive feature selection attempts should be avoided.

Overall, in terms of adaptability to cost-sensitive spam filtering, the classifiers based on LSI and RBF demonstrate their strength in this evaluation. Although these are two quite different machine-learning algorithms, they share one common characteristic: that is, both use a clustering component in their model training. Since clustering can potentially group messages by topics, an integrable clustering process can benefit from machine-learning algorithms in enhancing their profile accuracy of legitimate email (i.e. the category with a large vocabulary), and in reducing their numbers of false positive errors.

3.7 Concluding remarks

In this chapter, we provide an evaluation study of five current machine-learning algorithms proposed for spam filtering. The algorithms are described and compared by using various feature sizes, determined through an effective feature selection procedure, and by conducting experiments on some benchmark spam testing corpora constructed from two different languages. In particular, this study evaluates the adaptability of the algorithms for cost-sensitive spam filtering and, in this regard, the classifiers based on augmented LSI spaces, SVM, and RBF networks are the top performers. The experimental results also suggest that the newly proposed LSI and RBF classifiers represent two very competitive alternatives to other well-known methods for text and spam classification.

Content-based spam email filtering is a challenging classification task and success of the process can practically be influenced by many choices that include the selection of the algorithm, data and data preprocessing, feature selection, and decision criteria. In this study, we use only the email subject line and body text as the content for learning. For future work, we plan to expand the email content for spam filtering by the features contained in header fields, which seem to be reliable and useful (Zhang et al. 2004). Also, we plan to revisit some machine-learning algorithms to further improve their classification effectiveness on cost-sensitive learning. For instance, we would like to see how an optimal number of clusters for the LSI and RBF classifiers can be determined to create an accurate representation of topics among messages of both email categories.

3.8 Acknowledgements

This work was in part supported by a faculty research grant from the University of San Diego.

References


