Spam classification: a comparative analysis of different boosted decision tree approaches

Shrawan Kumar Trivedi
Indian Institute of Management Sirmaur, Sirmaur, India, and
Prabin Kumar Panigrahi
Indian Institute of Management Indore, Indore, India

Received 2 November 2017
Revised 9 March 2018
15 March 2018
21 March 2018
24 May 2018
Accepted 26 July 2018

Abstract

Purpose – Email spam classification is now becoming a challenging area in the domain of text classification. Precise and robust classifiers are not only judged by classification accuracy but also by sensitivity (correctly classified legitimate emails) and specificity (correctly classified unsolicited emails) towards the accurate classification, captured by both false positive and false negative rates. This paper aims to present a comparative study between various decision tree classifiers (such as AD tree, decision stump and REP tree) with/without different boosting algorithms (bagging, boosting with re-sample and AdaBoost).

Design/methodology/approach – Artificial intelligence and text mining approaches have been incorporated in this study. Each decision tree classifier in this study is tested on informative words/features selected from the two publically available data sets (SpamAssassin and LingSpam) using a greedy step-wise feature search method.

Findings – Outcomes of this study show that without boosting, the REP tree provides high performance accuracy with the AD tree ranking as the second-best performer. Decision stump is found to be the under-performing classifier of this study. However, with boosting, the combination of REP tree and AdaBoost compares favourably with other classification models. If the metrics false positive rate and performance accuracy are taken together, AD tree and REP tree with AdaBoost were both found to carry out an effective classification task. Greedy stepwise has proven its worth in this study by selecting a subset of valuable features to identify the correct class of emails.

Research limitations/implications – This research is focussed on the classification of those email spams that are written in the English language only. The proposed models work with content (words/features) of email data that is mostly found in the body of the mail. Image spam has not been included in this study. Other messages such as short message service or multi-media messaging service were not included in this study.

Practical implications – In this research, a boosted decision tree approach has been proposed and used to classify email spam and ham files; this is found to be a highly effective approach in comparison with other state-of-the-art modes used in other studies. This classifier may be tested for different applications and may provide new insights for developers and researchers.

Originality/value – A comparison of decision tree classifiers with/without ensemble has been presented for spam classification.

Keywords Adaboost, Bagging, Boosting, Decision tree classifiers, Greedy stepwise feature search

Paper type Research paper

1. Introduction

The business environment in the modern day world is highly competitive with little scope for complacency. In the context of an open system of communications, the exchange and sharing of information has become indispensable for organisations. A seamless exchange of information amongst organisations leads to better understanding and value creation. At the

Journal of Systems and Information Technology
Vol. 20 No. 3, 2018
pp. 298-320

1328-7265
DOI 10.1108/JSIT-11-2017-0105

same time, all forms of communication should be cost-effective and affordable. The cheapest method of exchanging electronically stored messages between people is electronic mail or "*email*". With email messages, it is possible to send text files as well as non-text files such as audio and image files. Emails can be exchanged over the internet as well as over public and private networks.

One of the problems with email usage is unsolicited messages sent over the internet; this is known as spam. Typically, entities use spam to send messages to a large number of email users for the purposes of advertising, phishing or accessing information. By sponsoring advertisements through e-mail spam, companies can earn huge money. As a consequence, email users face different types of problems when they receive unsolicited messages that can be harmful and unwanted.

According to recent research, the proportion of spams and their occurrence on the internet, especially in the form of emails, has risen to a staggering 70 per cent of the entire worldwide flow of emails (Aladin Knowledge System). These spams occupy a major portion of the available mailbox space, resulting in significant time being wasted to remove them (Lai, 2007). The thin demarcation between a legitimate or unwanted email creates the necessity for a precise and robust spam filter.

The expertise of spammers makes the issue more challenging. Spammers use various methods to send spams that are a challenge to anti-spam systems. One of the methods used by spammers is "tokenization", where the structure or the content of the email is fragmented into multiple parts or divisions such as "three" written as "thr33" or adding an HTML link to the email. The email itself is very different from any legitimate information sent to the user (Trivedi and Dey, 2013a).

In this paper, a comparative study of different decision tree classifiers that classify emails into spam (unsolicited) or ham (legitimate) has been carried out. Using two data sets available on the public domain, all the decision tree classifiers are tested and compared with/without 'boosting algorithm.' After comparison, a precise and robust decision tree model with boosting has been successfully formed using the email spam classification applications.

In current literature, decision tree classifiers have a prominent place. Sometimes, it is found to be more user-friendly than other popular classifiers such as neural network and support vector machines due to its capability to tackle the presentation of data in an efficient way. A number of approaches to develop a decision tree have been successfully proposed (Quinlan, 1993). Another important feature of decision tree is its flexibility in handling real-value and categorical attributes as well as items with missing attributes. In addition, decision trees have the capacity to classify more than two class problems efficiently and can be modified to deal with regression problems (Kingsford and Salzberg, 2008).

This study recommends a decision tree classifier in association with boosting algorithms; the overall delivery of the classifier makes for better precision and efficiency. The respective classifier uses the technique and voting mechanism to elevate performance.

The rest of the paper is presented as follows. Section 2 discusses the related work. Section 3 covers the machine learning classifier. Section 4 discusses experimental design. Section 5 discussed Results and findings followed by a conclusion in section 6.

2. Related work

Spam classification is a challenging area because of the smart activities of spammers; they frequently alter the spam words to introduce different forms of attack. A number of machine learning based classifiers have been highlighted in the literature to counter the above attacks. This section shows some existing work related to this research.

Decision trees (Carreras and Marquez, 2001) have been widely implemented in the research of classification. A decision tree constructs some observations about instances to draw a classification decision. For training, it takes one observation at a time to split the data. It chooses the order in which to examine observations.

Rios and Zha (2004) designed a random forest (RF) classifier to test on a time-indexed data that incorporates text and metadata features. They have observed from this study that RF was as good as SVM in terms of false positive rate.

A boosting approach has also been used in spam classification research. Sakkis *et al.* (2001) have produced a combining classifier to generate an improved ensemble. This classifier includes Naive Bayes (NB) and kNN classifiers to obtain best classification results. Another piece of research by Carreras *et al.* (2003) predict that boosted decision tree performs better than a decision tree classifier alone. A study by Trivedi and Dey (2013b) demonstrates boosting approaches and studies their effect on probabilistic classifiers; they identified that this method helps classification by boosting performance even with a small subset of informative features. Datta *et al.* (2015) worked on video traffic classification with j48 and AdaBoost, where they carried out experiments on Google hangout data with 2.5 packets collection; this achieved 99.99 per cent accuracy. Gashti (2017) examined spam email classification with decision tree (CART), SVM, NB and multi-layer perception (MLP), achieving 87.05 – 100.00 per cent accuracy for the CART algorithm. Conversely, work done by Shah and Kumar (2018) incorporates decision tree (Id3) and other machine learning; this shows accuracy up to 83.92 per cent

3. Machine learning classifiers

3.1 Decision tree

A simple decision tree classifier (Trivedi and Dey, 2013c) is based on the C4.5 algorithm. The C4.5 algorithm selects the most informative feature from the set of features. The feature is selected by normalising the information gain (i.e. entropy difference). This algorithm follows some base cases:

- If the entire sample belongs to a similar category, a leaf node in the decision tree is constructed to select that class.
- If none of the features offer information gain, it develops a decision node higher up in the tree by calculating the expected value of the class.
- If the example of a previous untouched class is encountered, it again creates a decision node higher up the tree by calculating the expected value of the class.

Algorithm for C4.5:

- Verification of base cases
- For each feature x^i, calculate normalise information gain.
- For the most informative feature x_b^i (higher normalise gain), develop a decision node that splits onx_b^i.
- Revise above steps on the sub lists constructed from splitting on x_b^i.

In this paper, three different kinds of decision tree classifiers are used:

3.1.1 Alternating decision tree. An alternating decision tree (Freund and Mason, 1999) is a machine learning classifier which works by generalising the decision tree and has a connection with boosting. This decision tree classifier is constructed from decision and prediction nodes. In addition, the decision nodes assume some predicate condition whereas

prediction nodes carry a single number. The prediction nodes are found in both the roots and leaf. The classification mechanism is constructed by following each path for all decision nodes that are fulfilled and then adding any prediction nodes that are traversed.

3.1.2 Decision stump. This decision tree classifier (Iba and Langley, 1992) is a one level decision tree because it carries only one internal node which connects to the terminal node (leaves). Decision stump works by developing prediction that depends on the value of a single input feature. This classifier is also known as the 1-rule classifier.

3.1.3 Reduced error pruning (REP tree). This decision tree (Quinlan, 1987; Witten and Frank, 2005) works on the regression tree logic and constructs multiple trees iteratively. Among all generated trees, it selects the best one to represent whole trees. For pruning, it calculates the mean square error on the predictions produced by the trees.

3.2 Boosting algorithms

"Boosting" is a method which works on improving the performance of any machine learning algorithms. Theoretically, this method is used to minimise the learning error of a "weak" Zhou (2012) learning algorithm and focuses on building an ensemble of the many classifiers which are weak (unable to handle data sample efficiently) algorithms; this improves results in a logical manner rather than random guessing (Freund and Schapire, 1996).

The concept of boosting algorithms is taken from the bootstrapping method (Duda *et al.*, 2001; Efron, 1982). Bootstrapping is used to assess the statistical accuracy of estimates. It works on a sampling-based statistical method where the sample is drawn with replacement from data points set. For the various applications of classification, some boosting algorithms have performed well with significant results. Different kinds of boosting algorithms have been used in this research as follows:

3.2.1 Bagging (Bag). A bagging technique (Duda *et al.*, 2001; Breiman, 1996; Friedman *et al.*, 2001) works on the bootstrapping concept and is used to classify data from various applications. After the generation of individual bootstraps, it aggregates them. Let us consider a training set $T = \{t_1, t_2 \ldots t_n\}$ with $t_i = (x_i, y_i)$. This algorithm focuses on fitting a regression model to develop a prediction f_x^b at the input x. By averaging the prediction over the set of bootstraps, it reduces the variance and increases the accuracy. In this paper, the model is fitted by prediction f_x^b for each bootstrap sample T^b with $b = 1, 2, 3, \ldots B$. The bagging estimation can be defined by equation (1):

$$f_x^{bag} = \frac{1}{B}\sum_{b=1}^{B} f_x^b \tag{1}$$

Algorithm for bagging:

Input: Training set $T = \{t_1, t_2 \ldots t_n\}$ with $t_i = (x_i, y_i)$. Number of sample version of training set B

Output: A boosted classifier with the training set G_x.

For $n = 1$ to B

- drawing with replacement $K \leq$ N sample from the training set T, taking the n^{th} sample T^n;
- training of the classifier G_n, for each sample T^n; and
- produce the final classifier as a vote of G_n with $n = 1$ to B.

$$G_x = sign\left(\sum_{n=1}^{B} G_n^x\right) \tag{2}$$

3.2.2 Boosting (Resample method). Boosting with re-sample technique works in a similar way to bootstrap and bagging. The only difference is the bootstrapping and bagging techniques use sampling with the replacement while boosting with re-sample performs sampling without replacement. This technique was introduced by Schapire (1990).

Algorithm for classification:

Input: Training set $T = \{t_1, t_2 \ldots t_n\}$ *with* $t_i = (x_i, y_i)$. Number of sample version of training set B

Output: A boosted classifier for the training set G_x.

- Drawing without replacement $K^1 < N$ sample out of the training set T to find the sample T^1.Training of the weak classifier G_1, for each sample T^1.
- Choose $K^2 < N$sample from the training set T together with half of misclassified sample obtained by G_1. Training of weak classifier G_2 on it.
- Choose all other samples misclassified by G_1 and G_2 and train weak classifier G_3 on it.
- Build the final boosted classifier based on the voting of weak classifiers

$$G_x = sign\left(\sum\nolimits_{n=1}^{3} G_n^x\right) \tag{3}$$

3.2.3 Adaptive boosting (AdaBoost). AdaBoost (Friedman *et al.*, 2001; Freund and Schapire, 1996) uses the concept of reweighting the data rather than random sampling. It works on the concept of building the ensemble for the classifiers to improve the performance accuracy. This algorithm combines the output M of the weak classifierG_m^x, for the final classification decision carried out by $G_x = sign\left(\theta_m \sum\nolimits_{m=1}^{M} G_m^x\right)$.

Algorithm for classification:

Input: Training set $T = \{t_1, t_2 \ldots t_n\}$ with $t_i = (x_i, y_i)$. Number of sample version of training set B

Output: A boosted classifier for the training set G_x.

(1) Initialise the weights $w_i^t = \frac{1}{N}, i = \{1, 2, 3, \ldots N\}$.

(2) For $m = 1, 2, 3, \ldots M$.

- Learn the classifier G_m^x from the training data according to their weights w_i^t.
- Compute error term $E_m^{rr} = \frac{\sum_{i=1}^{N} w_i^t I(y_i \neq G_m^{x_i})}{\sum_{i=1}^{N} w_i^t}$
- Evaluate weight contribution.
- $\theta_m = 0.5\log\left(\frac{1-E_m^{rr}}{E_m^{rr}}\right)$.
- Substitute $w_i^t \leftarrow w_i^t Exp\left(-\theta_{(m)} I\left(y_i \neq G_m^{x_i}\right)\right)$ then renormalize $\sum_i w_i^t = 1$.

(3) The final boosted classifier

$$G_x = \theta_m sign\left(\sum_{m=1}^{M_x} G_m^x\right) \quad (4)$$

4. Experimental design

4.1 Data sets

4.1.1 SpamAssassin. The main data set of this study is SpamAssassin (Trivedi and Dey, 2013a, 2013b, 2013c, 2014). This data set includes some older and some recent spam emails produced by some non-spam-trap sources. From the entire spam outsets, 2350 spam email files are taken for this study. This data set also has some easy and hard legitimate files. To balance the rate, easy and hard email files are mixed together to produce 2,350 ham email files. Easy and hard ham emails are distinguished by the complexity imbibed on them. Easy emails can be simply identified and classified, whereas hard ham emails are affected by some attacks (such as imbibe HTML code, unusual HTML markup, different coloured text, Spammish-sounding phrase on the features) carried out by spammers; this makes it difficult to identify such emails.

4.1.2 LingSpam. The second data set is taken from the LingSpam corpus. This corpus includes four different versions of email files such as bare (Lemmatiser disabled and stop-word disabled), lemme (Lemmatiser enabled and stop-word disabled), lemm_stop (Lemmatiser enabled and stop-word enabled) and stop (Lemmatiser disabled and stop-word enabled). For this study, 478 spam files and 478 ham files, including all versions of LingSpam corpus, is chosen (Sahami et al., 1998; Cranor and LaMacchia, 1998).

4.2 Pre-processing of data set

Pre-processing (Sergienko *et al.*, 2017) (Figure. 1, Table II) is performed to transform the incoming email files where strings of characters are transformed to a representation suitable to the classification algorithm.

Figure 1 clearly depicts the structure of a spam classifier where an email file with header and body undergoes the process of pre-processing. In this research, only the body part has been included as this part of the email has the main content. In the pre-processing phase, tokenisation (extraction of the words from the email file), stop word removal (removal of the words that often occur in the email files such as "articles", "prepositions" etc.) and lemmatisation (reducing word to its normal form, such as "boosting, boosted" may be reduced to "boost" only) are carried out (Patel and Patel, 2017). After the feature extraction process, feature selection is introduced where an informative feature of the words is selected. In this research, "greedy stepwise feature subset selection method" is involved. After gathering all of the informative features, feature representation is conducted. A binary feature representation method is involved to produce a term to documents matrix (TDM). Finally, a machine learning model is trained and validated with the classification output. The following section describes all the steps of a spam classifier in detail.

4.2.1 Feature extraction. In this process (Ramya *et al.*, 2017), the information from the email files is extracted to develop an associated feature dictionary. The string-to-word-vector mechanism is applied; this process includes HTML (or other) tags removal, stop-word removal (some words which appear very often such as articles, prepositions and conjunctions etc.) and lemmatisation (reducing words to their basic form such as improving to improve). The selected feature set is then taken through the dimensionality reduction process.

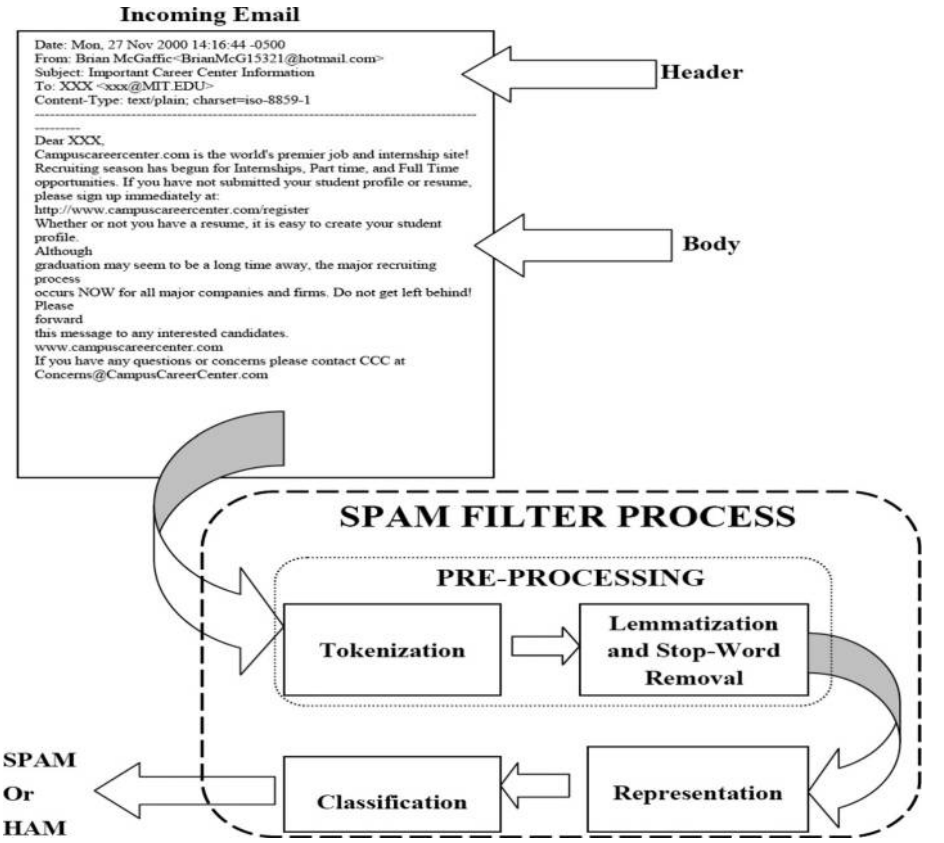

Figure 1.
Structure of spam classifier

4.2.2 Dimensionality reduction. The major problem in spam classification is high dimensionality (Trivedi and Dey, 2016a, 2016b, 2016c, 2016d) of the feature space, where one dimension of a unique word is found in many files. This large set of the feature space creates difficulty for standard classification methods, as the computation process is costly and also results are found to be unreliable. Therefore, this large feature space has to be reduced. This reduction process is known as dimensionality reduction and is enabled by a feature selection process.

4.2.2.1 Feature subset search. This technique (Trivedi and Dey, 2013c; Tripathi and Trivedi, 2016) is used to obtain the most informative feature subset aimed at reducing the feature space. Many feature selection mechanisms have been popularised in existing literature; these include wrapper feature subset, classifier subset evaluator, consistency subset evaluator, cost-sensitive subset evaluator, etc.

These methods basically use the feedback concept and machine learning algorithm to select the most informative feature subset. Various studies have examined different feature subset selection methods for searching out the most informative features along with machine learning classifiers. In this research, greedy step-wise search (Trivedi and Dey, 2013c) is considered.

S/N	Author(s) and year	Model used	Data source/corpus	Accuracy (%)
1	Drucker *et al.* (1999)	Boosting decision trees and other ML	Email data from AT&T Technical staff	Error rate1.8-2.78%
2	Carreras and Marquez (2001)	Decision Tree and other ML	PU1 corpus	F Value 89.25
3	Rios and Zha (2004)	Random Forest and SVM	60,000 Spam and 15000 Ham	NA
4	Ntoulas *et al.* (2006)	Decision Tree and other ML	Collection of 105, 484, 446 web pages by MSN search crawler	95.4
5	Yang *et al.* (2006)	Model based on embedded decision tree	LingSpam and SpamAssassin	MMR % = 0.90-1.76 (LingSpam), 3.08-4.03 (SpamAssassin)
6	Exarchos *et al.* (2007)	Decision Tree	Corpus from European Society of Cardiology ST-T database	92
7	Gadat and Younes (2007)	Stochastic gradient descent algorithm, Random forest	Leukaemia Database	Rate of misclassification = less than 0.05
8	Castillo *et al.* (2007)	Decision Tree	WEBSPAM-UK2006 corpus	F-Value 64.6
9	Koprinska *et al.* (2007)	Decision Tree, RF and other ML	Email from four users (U1-U4) and Authors email U5, Ling Spam, PU1 and U5Spam	68.32-96.03
10	Abu-Nimeh *et al.* (2007)	SVM and Various Tree-based classifiers	A corpus of 2889 phishing and legitimate emails	87.07-90.24
11	Youn and McLeod, (2007)	Ontology based Decision Tree	4600 emails 39.4% spam rate	97.17
12	Tang *et al.* (2008)	Random Forest and SVM	IPs From the Trusted source corpus	95.71
13	Janecek *et al.* (2008)	Decision Tree and other ML	TREC 2005 e-mail corpus and drug discovery corpus	99.7 for first data and 79.5 for second data
14	DeBarr and Wechsler, (2009)	Random Forests, and Active Learning	2007 TREC Public Spam Corpus	95.2
15	Abdulsalam *et al.* (2011)	Streaming Random Forests	Synthetic and real corpuses	Classification error % = 0.0-1.2
16	Erdélyi *et al.* (2011)	Ensemble selection, Logit Boost and Random Forest	WEB-SPAM-07, ECML/ PKDD Discovery Challenge corpus DC2010	3.5-5 % improvement
17	Kumar *et al.* (2012)	Decision Tree with other Machine Learning Model and different feature selection	Spam base dataset	90% to 95%
19	Zang *et al.* (2012)	Wrapper feature selection with Decision tree	5600 emails data set	Error of misclassifying non spams as spams is only 1%

(*continued*)

Table I. Research work related to tree-based algorithms

S/N	Author(s) and year	Model used	Data source/corpus	Accuracy (%)
20	Trivedi and Dey (2013a, 2013b, 2013c, 2013d)	Boosting approaches on Probabilistic classifiers	Enron Email data	88-92%
21	Zhang *et al.* (2014)	Decision Tree	6000 emails Dataset	91.02-94.27%
22	Jayetta Datta *et al.* (2015)	J48 with AdaBoost	Google hangout data	99.99%
23	Goyal *et al.* (2016)	kNN and Decision Tree	Datasets of twitter	74.8% for decision tree
23	Khodabandehlou and Zivari Rahman (2017)	Decision Tree and ANN with boosting	Customer behavioural data	ANN was good than Decision Tree (97.92)
24	Gashti, (2017)	decision tree (CART), SVM, NB and MLP	Spam Base, Ling Spam and PU1	87.05-100.00%
25	Shah and Kumar (2018)	ID 3 and other machine learning methods	10000 email data for training and 2000 email data for testing	78.57-83.92%

Table I.

1	Step	Submit	Taint	Thousands	Top	Total	Trading	Transaction
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	1
9	0	0	0	0	0	0	0	0
10	1	1	0	0	1	0	0	0
11	0	1	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0

Table II.
Document term binary representation

Let us consider f_s as the feature set for the search process and f_e the number of features under evaluation with respect to their fitness. Hence, the most informative feature wrapper set f_b^* is:

$$f_b^* = \arg\max_{f_e \notin f_s} fit(f_s \cup \{f_e\}) \tag{5}$$

4.2.3 Feature representation. An important component in text classification research is the features representation (Wu *et al.*, 2016; Tripathi and Trivedi, 2016). In email files, the text is generally obtained from the body of the message but sometimes the header and subject can also be considered. One of the popular representation techniques is the bag of words (BoW) (also known as vector space model). Some other approaches, such as character N-gram, are also used for representation but have not been practised extensively, particularly for spam classification.

Sometimes, selected features are represented by a binary representation method where email files and words together form a binary matrix, known as the term-document matrix (TDM). This method is known as the term weighting method. This binary matrix contains binary values (1 and 0), where 1 indicates the presence of the particular feature or word in a specific email file and 0 otherwise.

Term weighting method has been chosen for this research. Let us suppose that each email file is represented as a column vector D^x defined by the words extracted from the email files, i.e. $D^x = (w^1, w^2, w^3....)$ where w^i is termed as i^{th} word/feature of the email document d^x (Aas and Eikvil, 1999). The combination of all email documents and words form a $M \times N$ matrix where M represents the number of distinct features and N represents the number of email instances. Table III represents the term document relationship as a a^{ij} matrix that is defined as the degree of relationship between word i (column) and email file j (row).

4.3 Evaluation

To classify emails, a number of the most informative features were selected separately from two different datasets. Thereafter, data are split with 66 per cent data taken for training; the remaining 34 per cent of data is used to test the concerned classifiers. This study used three different performance measure metrics (Table IV) for evaluation and analysis purpose. The simplest instrument of this study is the performance accuracy, defined as the percentage of accurate classified emails files. The second metric, i.e. F-value is also recorded for accuracy; it is defined as the harmonic sum of precision (fraction of retrieved classified email files that are relevant) and recall (fraction of accurate classified email files that are retrieved); this is considered as the most informative measurement. The last important metric taken in this study is false positive rate that measures the sensitivity of accurate classification and tells "how many positive instances are misclassified."

Where $N_{Ham \to c}$ is the total number of correctly classified ham emails; $N_{Ham \to m}$ is the number of misclassified ham emails; $N_{Spam \to c}$ is defined as correctly classified spam emails and $N_{Spam \to m}$ is the total number of misclassified spam emails.

Tokenisation	After pre-processing	After feature selection
Delivered	Deliver	Spamassassin
Exim	Exim	Taint
Habeas	Habeas	Communication
For	Justin	Geneva
Habeas	business	Jalapeno
If	package	Users
Justin	cash	Laugh

Table III. Most informative feature selection process (SpamAssassin data set)

Instruments	Related formulas
Accuracy	$A_{ccuracy} = \frac{N_{Ham \to c} + N_{Spam \to c}}{N_{Ham \to c} + N_{Ham \to m} + N_{Spam \to c} + N_{Spam \to m}}$
F-Value	$F = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision+Recall}}$
False positive rate	$FP_{rate} = \frac{N_{Ham \to m}}{N_{Ham \to m} + N_{Ham \to c}}$

Table IV. Instruments for performance measure (Sebastiani, 1999)

A. Software:

MATLAB 2008 and JAVA-based environment on the WINDOW 7 operating system with 4GB RAM has been used for this research. In this study, all the classification models were tested with the help of JAVA-based tools. Classification results were taken for visualisation with the help of MATLAB 2008. This was available to us and has been used to make graphs only.

5. Results and discussion

The analysis is divided into three different sections. The first section demonstrates the analysis of concerned decision tree classifiers without boosting algorithms. The second part shows the analysis of classifiers with the use of various boosting algorithms. The last section demonstrates the analysis of combined effect. All analysis has been conducted with the use of three metrics (performance accuracy, F-value and false positive rate). However, the results of performance accuracy and F-value were similar to each other. All classifiers were tested on two publically available datasets where the 36 most informative features from the main dataset SpamAssassin and the 58 most informative features from the LingSpam data set (taken for validation purpose) were selected; the greedy step-wise feature subset selection method was used.

5.1 Analysis of decision tree classifiers (Without boosting)

5.1.1 With performance accuracy. Table V and Figure 2 demonstrate the performance of concerned decision tree classifiers (AD tree, decision stump and REP tree) without boosting, tested on the SpamAssassin data set. Results show that REP tree and AD tree give excellent performance accuracy (97.5 and 96.3 per cent, respectively). Decision stump is predicted to be an underperforming classifier with 87 per cent performance accuracy.

The same classifiers tested on the LingSpam data set (Table VI and Figure 3) validate the results emerging from the first data set. In this case, the results for REP tree and AD tree were more or less the same with performance accuracy of 96.1 and 96.3 per cent, respectively. Decision stump has again performed relatively poorly with 82.2 per cent accuracy.

5.1.2 With false positive rate. Table VII and Figure 4 show the false positive rate of the classifiers without boosting for the SpamAssassin data set. Overall, decision stump performed well in false positive rate (0 per cent), whereas the REP tree also gave a satisfactory result with 1.4 per cent FP rate. In addition, AD tree (with 2 per cent FP rate) is predicted to be worst in this case.

When the same classifiers were tested on the LingSpam data set (Table VIII and Figure 4), contradictory results are obtained. In this case, AD tree (with 1.3 per cent FP rate)

Table V. Accuracy and F-Value for Spam Assassin data set

	Decision tree classifiers					
	AD tree		Decision stump		REP tree	
Spam Assassin	Acc. (%)	F value (%)	Acc. (%)	F value (%)	Acc. (%)	F value (%)
BOOSTING						
Without Boost	96.3	96.3	87.2	87	97.5	97.5
Bag	97.1	97.1	87.2	87.1	97.8	97.8
Boosting	97.7	97.7	95.8	95.8	98.2	98.2
ABoost	98.1	98.1	94.4	94.3	98.3	98.3

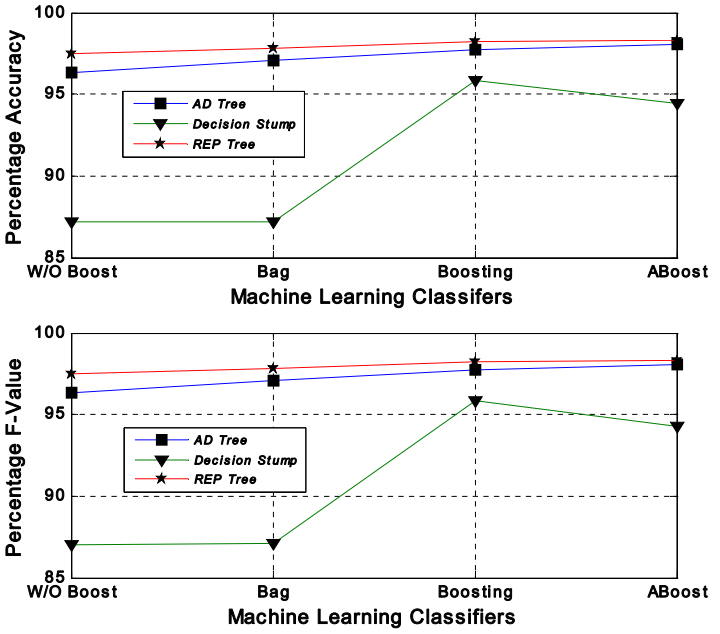

Figure 2. Accuracy and F-Value for LingSpam data set

	Decision tree classifiers					
	AD tree		Decision stump		REP tree	
LingSpam	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)
BOOSTING						
Without Boost	96.3	96.2	82.2	82.2	96.1	96
Bag	97.3	97.2	88.2	88.1	96.3	96.2
Boosting	97.8	97.8	96.3	96.3	97.8	97.7
ABoost	97.9	97.8	96.1	96	97.8	97.8

Table VI. Accuracy and *F*-Value for LingSpam data set

was predicted to be better, whereas the REP tree (with 1.9) was the second best classifier. Decision stump was worst (with 4.4 per cent FP rate) in this case.

5.2 Analysis of decision tree classifiers (With boosting)

5.2.1 With performance accuracy. From Table V and Figure 2, it is clear that boosting improves the performance accuracy of the classifiers. These observations were noted from classifiers tested on SpamAssassin data set; these demonstrate that REP tree and AD tree are both good classifiers with/without boosting with performance accuracy 96.3 to 98.1 per cent for AD tree and 97.5 to 98.3 per cent for REP tree. Decision stump is shown to be weak with performance accuracy 87.2 to 94.4 per cent.

The same classifiers were tested on the LingSpam data set. Table VI and Figure 3 confirm the results for the SpamAssassin data set. In this case, again REP tree and AD tree are the best classifiers with/without boosting; performance accuracy for REP tree is

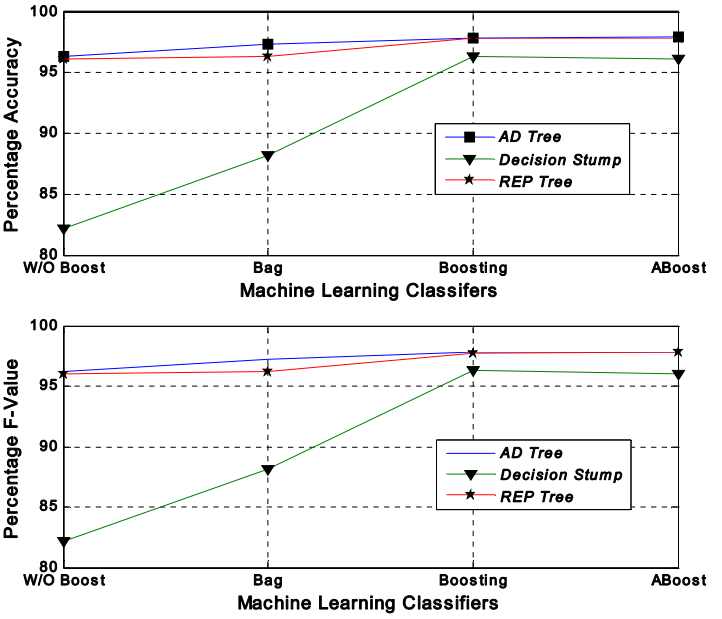

Figure 3.
Accuracy and F-Value for LingSpam data set

SpamAssassin FP rate (%)	Different decision tree classifiers AD Tree	Decision stump	REP tree
BOOSTING			
Without Boost	2	0	1.4
Bag	2	0	1
Boosting	1.2	2.7	1.4
ABoost	1.1	2.7	1.7

Table VII.
False positive rate for SpamAssassin data set

96.3 per cent to 97.9 per cent and 96.2 per cent to 97.8 per cent for AD tree. Decision stump is found to be underperforming again with 82.2 per cent to 96.1 per cent performance accuracy.

5.2.2 With false positive rate. In the case of boosting algorithms when tested on the SpamAssassin data set (Table VII and Figure 4) in conjunction with different decision tree classifiers, AdaBoost was best with 1.1 to 2.7 per cent FP rate.

Similar results were obtained from the LingSpam data set (Table VIII and Figure 4) where again AdaBoost performed well with 0.6 per cent to 2.6 per cent FP rate.

5.3 Analysis with combination of boosting and decision tree

5.3.1 With performance accuracy. Table V and Figure 2 demonstrate the effect of various boosting algorithms on the different decision tree classifiers tested on the SpamAssassin data set. Results show that AD tree and REP tree are both proven to be excellent classifiers with the use of AdaBoost and boosting with re-sample algorithms. AdaBoost was shown to be the best boosting algorithm, while boosting with re-sample was the second best

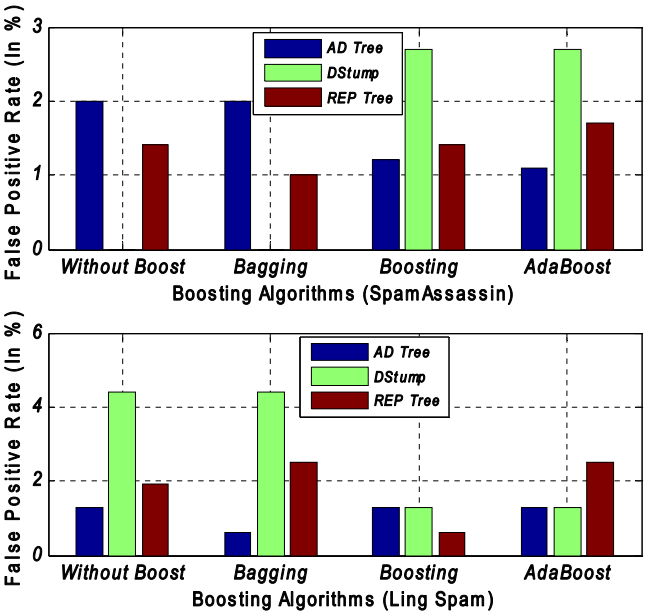

Figure 4.
FP rate for both dataset

SpamAssassin FP rate (%)	Different decision tree classifiers AD Tree	Decision stump	REP tree
BOOSTING			
Without Boost	1.3	4.4	1.9
Bag	0.6	4.4	2.5
Boosting	1.3	1.3	0.6
ABoost	1.3	2.6	2.5

Table VIII.
False positive rate for LinSpam data set

algorithm for boosting. REP tree with AdaBoost was identified to be a better combination with 98.3 per cent accuracy, whereas REP tree with boosting with re-sample was second best with 98.2 per cent performance accuracy; bagging is predicted to be a weak boosting algorithm.

Table VI and Figure 3 demonstrate the observations of the classifiers tested on the LingSpam data set that validates the SpamAssassin data set. In this case, the REP tree classifier with AdaBoost/boosting with re-sampling algorithms was best with 97.8 per cent accuracy. The bagging algorithm has again proven to be weak.

5.3.2 With false positive rate. In the view of combined effect (Table VII, Figure 4), when the same classifiers were tested with boosting algorithms on the SpamAssassin data set, REP tree in conjunction with AdaBoost (with 1.2 per cent FP rate) is predicted to be excellent, whereas REP tree (1 to 1.7 per cent FP rate) was the second best in all combinations. Decision stump classifier is again predicted to be worst.

On the other hand, the classifiers when tested on the LingSpam data set (Table VIII, Figure 4) validate perfectly the results emerging from the SpamAssassin data set. In this

case, AD tree with Adaboost again proves to be better with 1.3 per cent FP rate; REP tree with AdaBoost was second best.

5.4 Significance test (10 fold cross-validation of accuracy and F-value)

Cross-validation is a method of evaluating classification models by splitting the corpus into a training set to train the model, and a test set to evaluate it. In 10-fold cross-validation (Moreno-Torres, *et al.*, 2012), the corpus is randomly split into ten equal size subsamples. Of the ten subsamples, a single subsample is taken as the validation data for testing the model, with the remaining nine subsamples used as training data. The cross-validation process is then repeated ten times (the folds), with each of the ten subsamples used exactly once as the validation data. The ten results from the folds can then be averaged to get a single estimation. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

5.4.1 Analysis of accuracy and F value with 10-fold cross-validation. After cross-validation, the results of accuracy and f-value of experiment setting 66-34 per cent training and testing (Table V and VI; Figures 2, 3) and 10-fold cross-validation (Tables IX, X) have been compared. Results of 10-fold cross-validation strongly support the results from the first experiment setting for both datasets. In the validation process, again the REP decision tree with Adaboost has been predicted as the most accurate model among those tested in this study with 99.1 per cent accuracy for the SpamAssassin data set and 98.2 per cent for the Lingspam data set.

5.4.1.1 Discussion and comparison. Table XI shows the comparative analysis of decision tree classifiers in recent work. The work done by Trivedi and Dey (2013d) uses decision tree classifiers and builds a model for Enron dataset with 88-92 per cent accuracy. If we compare the proposed boosted decision tree model, it clearly depicts that the proposed model

Table IX.
Accuracy and *F*-Value for SpamAssassin data set

	Decision tree classifiers					
	AD tree		Decision stump		REP tree	
LingSpam	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)
BOOSTING						
Without Boost	97.1	97.1	88.7	88.7	98.3	98.3
Bag	97.8	97.8	89.1	89.1	98.4	98.4
Boosting	98.2	98.2	96.4	96.4	99.1	99.1
ABoost	98.3	98.3	95.2	95.2	99.1	99.1

Table X.
Accuracy and *F*-Value for LingSpam data set

	Decision tree classifiers					
	AD tree		Decision stump		REP tree	
LingSpam	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)	Acc. (%)	*F*-value (%)
BOOSTING						
Without Boost	96.8	96.8	83.4	83.4	97.2	97.1
Bag	97.9	97.9	89.1	89.1	97.5	97.5
Boosting	98.0	98.0	97.2	97.2	98.2	98.2
ABoost	98.1	98.1	97.1	97.1	98.2	98.2

S/N	Authors	Methods	Data sets	Accuracy in %
1	Trivedi and Dey (2013a, 2013b, 2013c, 2013d)	Boosting approaches on Probabilistic classifiers	Enron Email data	88-92%
2	Zhang *et al.* (2014)	Decision Tree	6000 emails Dataset	91.02-94.27%
3	Jayetta Datta *et al.* (2015)	J48 with AdaBoost	Google hangout data	99.99%
4	Goyal *et al.* (2016)	kNN and Decision Tree	Datasets of twitter	74.8% for decision tree
5	Khodabandehlou and Zivari Rahman (2017)	Decision Tree and ANN with boosting	Customer behavioural data	ANN was good than Decision Tree (97.92)
6	Gashti (2017)	decision tree (CART), SVM, NB and MLP	Spam Base, Ling Spam and PU1	87.05-100.00%
7	Shah and Kumar (2018)	ID 3 and other machine learning methods	10,000 email data for training and 2,000 email data for testing	78.57-83.92%
8	*Proposed work Boosting Decision Tree*	Decision Tree with Boosting	Spamassassin (4,700 emails), Ling Spam (956 emails)	99.1 (for Spamassassin), 98.2% (for Ling Spam)

Table XI. Comparison with other work

compares favourably to Trivedi and Dey (2013d). The proposed model may also be tested on Enron data set. On the other hand, the proposed boosted decision tree model performs well in comparison to the work conducted by Zhang *et al.*(2014), Goyal *et al.* (2016), Khodabandehlou and Zivari Rahman (2017) as well as Shah and Kumar (2018). The proposed model may also be validated on the data set used in the studies mentioned. If the comparison of the proposed model is made with the work of Gashti, (2017) – a model with accuracy 87.05 to 100 per cent – our model is comparable with accuracy 98.2 to 99.1 per cent. After comparing the results of the proposed model with others in the field, it is evident that the proposed model shows promise in detecting spams from emails. However, the proposed model may be tested with other data sets.

5.5 Wilcoxon signed-rank test (For matched dataset)

This study uses the Wilcoxon signed-rank test (Hu *et al.*, 2016) for further verification of the predictive accuracy of the decision tree classifiers used. The Wilcoxon signed-rank test is used to test whether the outcome of the two classifiers is significantly different or not (Demšar, 2006). The null hypothesis for this setup would be "the predictive capability of two machine learning classifiers is the same"; hence, the mean difference of the accuracy of the classifiers would be zero. In this study, all the machine learning classifiers are tested on two different data sets. At first "Wilcoxon signed-rank test" is performed on the SpamAssassin dataset and then on the Lingspam data set. Table XII and Table XIII show the p-values of the pair-wise machine learning classifiers based on the values of F-measures for the SpamAssassin and Lingspam data sets, respectively. In Tables XII and XIII, the p-values that are greater than the significant level (i.e. $p > 0.05$) have been illustrated in bold. Four observations are noted from the tables:

(1) *Observation 1*: In terms of F-measure of the two data sets, most of the pair-wise decision tree classifiers are significantly different (p-value < 0.05) from the others with only a small number of exceptions (p-value > 0.05).

(2) *Observation 2*: REP tree and AD tree with Adaboost and boosting with re-sample methods perform well, significantly different from the decision stump classifier. The predictive capability of AD tree and REP tree with Adaboost was comparable.

Table XII.
Wilcoxon's signed-rank test between each pair OF classifiers on the spamassassin dataset (F-measure)

P-values*	Decision stump	REP tree	AD tree+ Bagging	Decision stump+ Bagging	REP tree+ Bagging	AD tree+ Boosting	Decision stump+ Boosting	REP tree+ Boosting	AD tree+ Adaboost	Decision stump+ Adaboost	REP tree + Adaboost
AD Tree	3.01E-11	*0.050092*	0.004317	3.01E-11	0.251828	0.00195	7.68E-08	2.14E-10	4.17E-09	3.01E-11	1.28E-09
Decision Stump		3.01E-11	3.00E-11	0.002819	3.00E-11	3.00E-11	3.00E-11	3.00E-11	3.01E-11	3.01E-11	3.01E-11
REP Tree			0.00007	3.01E-11	*0.46873*	*0.58941*	5.44E-09	0.000051	0.00095	3.01E-11	0.001902
AD Tree + Bagging				3.01E-11	1.10E-08	1.76E-09	0.000005	3.00E-11	3.01E-11	3.01E-11	3.01E-11
Decision Stump + Bagging					3.01E-11	3.01E-11	1.85E-13	1.85E-13	1.85E-13	1.85E-13	1.85E-13
REP Tree + Bagging						1.83E-13	1.85E-13	1.85E-13	1.85E-13	1.85E-13	1.85E-13
AD Tree + Boosting							1.78E-09	0.02096	*0.0829*	1.78E-09	*0.081979*
Decision Stump + Boosting								3.00E-11	3.01E-11	3.51E-07	3.01E-11
REP Tree + Boosting									0.09184	3.00E-11	*0.050883*
AD Tree + Adaboost										3.01E-11	*0.830232*
Decision Stump + Adaboost											3.01E-11

Note: * Wilcoxon's Signed-Rank Test is performed on the 95% confidence level

P-values*	Decision stump	REP tree	AD tree+ Bagging	Decision stump+ Bagging	REP tree + Bagging	AD tree + Boosting	Decision stump+ Boosting	REP tree+ Boosting	AD tree+ Adaboost	Decision stump+ Adaboost	REP tree + Adaboost
AD Tree	2.83E-11	*0.410136*	0.002308	2.83E-11	0.028777	2.83E-11	*0.060378*	2.84E-11	2.87E-11	0.029507	3.13E-11
Decision Stump		2.89E-11	2.87E-11	2.85E-11	2.83E-11	2.86E-11	2.88E-11	2.87E-11	2.90E-11	2.83E-11	1.77E-13
REP Tree			0.035407	2.88E-11	0.007232	2.89E-11	*0.593604*	2.89E-11	2.93E-11	*0.485631*	1.11E-09
AD Tree+ Bagging				2.86E-11	0.104225	4.10E-10	0.000074	3.93E-09	1.88E-10	0.000038	0.000352
Decision Stump + Bagging					2.82E-11	2.85E-11	2.88E-11	2.86E-11	2.89E-11	2.82E-11	2.86E-11
REP Tree + Bagging						3.12E-11	0.000262	2.98E-11	2.87E-11	0.000052	2.77E-08
AD Tree + Boosting							2.88E-11	*0.116026*	*0.057867*	2.83E-11	0.000015
Decision Stump + Boosting								2.89E-11	2.92E-11	0.010542	4.09E-11
REP Tree + Boosting									0.001905	2.83E-11	0.001375
AD Tree + Adaboost										2.87E-11	3.27E-07

Note: * Wilcoxon's signed-rank test is performed on the 95% confidence level

Table XIII.
Wilcoxon's signed-rank test between each pair OF classifiers on the LingSpam data set (F-measure)

(3) *Observation 3*: The combination of REP tree and AD tree with/without boosting are found to be comparable with (p-value > 0.05). This indicates that the strength of both classifiers with/without boosting is significant.

(4) *Observation 4*: Decision stump classifier with/without boosting is found to be the underperforming classifier of this research. In all cases, the AD tree and REP tree with/without boosting are significantly different from the decision stump classifier with/without boosting.

6. Conclusion

The impact of spam (unsolicited) emails poses a number of challenges for organisations and huge potential financial losses. Some machine learning classifiers lack effective performance due to bad sampling and weak learning. With these challenges in mind, this research has studied the performance of various decision tree classifiers (AD tree, decision stump and REP tree) tested on two publically available data sets (SpamAssassin and LingSpam). REP tree and AD tree have performed well, with REP tree proving to be the best. Furthermore, different boosting algorithms (bagging, boosting with re-sample, AdaBoost) are incorporated to enhance the decision tree classifiers. Among all decision tree classifiers, REP tree and AD tree both perform well with AdaBoost and boosting with re-sampling; REP tree with AdaBoost is also found to be a good combination. Greedy step-wise search shows promise in searching out the most informative features of the datasets.

In addition, to validate the performance of the decision tree classifiers, 10-fold cross-validation of accuracy and F-measure has been conducted. In the validation process, the results show strong support to the results obtained from the 66-34 per cent training and testing split setup.

Furthermore, to check the significant difference of the performance accuracy and F-measure of decision tree classifiers, the Wilcoxon signed-rank test of matched data was carried out for both datasets. The results of this test suggest that most of the decision tree classifiers are significantly different on 95 per cent confidence interval (p-value < 0.05) from others with only a small number of exceptions (p-value > 0.05). Finally, this research concludes that REP tree and AD tree with Adaboost algorithms and boosting with resample are effective in classifying spam and ham emails.

This study imposes some limitations as well. Only English language emails have been used in this study. Neither sort message services nor multimedia services were incorporated. Image spam and attachments were not considered for this research. Training and testing time were not taken into account. To enhance this model, we are suggesting some future work as the follow up to this study.

In future, these boosting algorithms may be tested with other machine learning classifiers. The same study may be performed on other data sets. As this study concerns only email spam classifiers, the same proposed models may be used for other machine learning applications such as sentiment analysis, blog mining, news mining and other data and text mining applications. This research has taken only the body part of emails for analysis. Both header and body parts may be taken together for further research. Another limitation of this study is training time that has not been addressed. In future, the same models may be tested with other metrics such as training and testing time and ROC measure.

References

Aas, K. and Eikvil, L. (1999), *Text categorisation: A survey*.

Abdulsalam, H., Skillicorn, D.B. and Martin, P. (2011), "Classification using streaming random forests", *Ieee Transactions on Knowledge and Data Engineering*, Vol. 23 No. 1, pp. 22-36.

Abu-Nimeh, S., Nappa, D., Wang, X. and Nair, S. (2007), "A comparison of machine learning techniques for phishing detection", In *Proceedings of the Anti-phishing Working Groups 2nd Annual eCrime Researchers Summit*, *ACM*, pp. 60-69.

Breiman, L. (1996), "Bagging predictors", *Machine Learning*, Vol. 24 No. 2, pp. 123-140.

Carreras, X., Màrquez, L. and Padró, L. (2003), "A simple named entity extractor using AdaBoost", In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, *Association for Computational Linguistics*, Vol. 4, pp. 152-155.

Carreras, X. and Marquez, L. (2001), "Boosting trees for anti-spam email filtering", arXiv preprint cs/0109015.

Castillo, C., Donato, D., Gionis, A., Murdock, V. and Silvestri, F. (2007), "Know your neighbors: Web spam detection using the web topology", In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, *ACM*, pp. 423-430.

Cranor, L.F. and LaMacchia, B.A. (1998), "Spam!", *Communications of the Acm*, Vol. 41 No. 8, pp. 74-83.

Datta, J., Kataria, N. and Hubballi, N. (2015), "Network traffic classification in encrypted environment: a case study of google hangout", In *Communications (NCC), 2015 Twenty First National Conference on*, *IEEE*, pp. 1-6.

DeBarr, D. and Wechsler, H. (2009), "Spam detection using clustering, random forests, and active learning", In *Sixth Conference on Email and Anti-Spam. Mountain View*, *CA*.

Demšar, J. (2006), "Statistical comparisons of classifiers over multiple data sets", *Journal of Machine Learning Research*, Vol. 7, pp. 1-30.

Drucker, H., Wu, D. and Vapnik, V.N. (1999), "Support vector machines for spam categorization", *Ieee Transactions on Neural Networks*, Vol. 10 No. 5, pp. 1048-1054.

Duda, R.O., Hart, P.E., and Stork, D.G. (2001), "Pattern classification", *International Journal of Computational Intelligence and Applications*, Vol. 1, pp. 335-339.

Efron, B. (1982), "The jackknife, the bootstrap, and other resampling plans", *Siam*, Vol. 38.

Erdélyi, M., Garzó, A. and Benczúr, A.A. (2011), "Web spam classification: a few features worth more", In *Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality*, *ACM*, pp. 27-34.

Exarchos, T.P., Tsipouras, M.G., Exarchos, C.P., Papaloukas, C., Fotiadis, D.I. and Michalis, L.K. (2007), "A methodology for the automated creation of fuzzy expert systems for ischaemic and arrhythmic beat classification based on a set of rules obtained by a decision tree", *Artificial Intelligence in Medicine*, Vol. 40 No. 3, pp. 187-200.

Freund, Y. and Mason, L. (1999), "The alternating decision tree learning algorithm", *In ICML*, Vol. 99, pp. 124-133.

Freund, Y. and Schapire, R.E. (1996), "Experiments with a new boosting algorithm", *In Icml*, Vol. 96, pp. 148-156.

Friedman, J., Hastie, T. and Tibshirani, R. (2001), *The Elements of Statistical Learning*, Springer series in statistics, New York, NY, Vol. 1 No. 10.

Gadat, S. and Younes, L. (2007), "A stochastic algorithm for feature selection in pattern recognition", *The Journal of Machine Learning Research*, Vol. 8, pp. 509-547.

Gashti, M.Z. (2017), "Detection of spam email by combining harmony search algorithm and decision tree. Engineering", *Technology and Applied Science Research*, Vol. 7 No. 3, pp. 1713-1718.

Goyal, S., Chauhan, R.K. and Parveen, S. (2016), "Spam detection using KNN and decision tree mechanism in social network", In *Parallel, Distributed and Grid Computing (PDGC), 2016 Fourth International Conference on*, *IEEE*, pp. 522-526.

Hu, Z., Chiong, R., Pranata, I., Susilo, W. and Bao, Y. (2016), "Identifying malicious web domains using machine learning techniques with online credibility and performance data", *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 5186-5194.

Iba, W. and Langley, P. (1992), "Induction of One-Level decision trees", *In ML*, (pp. 233-240.

Janecek, A., Gansterer, W.N., Demel, M. and Ecker, G. (2008), "On the relationship between feature selection and classification accuracy", *Journal of Machine Learning Research-Proceedings Track*, Vol. 4, pp. 90-105.

Khodabandehlou, S. and Zivari Rahman, M. (2017), "Comparison of supervised machine learning techniques for customer churn prediction based on analysis of customer behavior", *Journal of Systems and Information Technology*, Vol. 19 Nos 1/2, pp. 65-93.

Kingsford, C. and Salzberg, S.L. (2008), "What are decision trees?", *Nature Biotechnology*, Vol. 26 No. 9, pp. 1011-1013.

Koprinska, I., Poon, J., Clark, J. and Chan, J. (2007), "Learning to classify e-mail", *Information Sciences*, Vol. 177 No. 10, pp. 2167-2187.

Kumar, R.K., Poonkuzhali, G. and Sudhakar, P. (2012), "Comparative study on email spam classifier using data mining techniques", In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, Vol. 1, pp. 14-16.

Lai, C.C. (2007), "An empirical study of three machine learning methods for spam filtering", *Knowledge-Based Systems*, Vol. 20 No. 3, pp. 249-254.

Moreno-Torres, J.G., Sáez, J.A. and Herrera, F. (2012), "Study on the impact of partition-induced dataset shift on $k $-fold cross-validation", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 23 No. 8, pp. 1304-1312.

Ntoulas, A., Najork, M., Manasse, M. and Fetterly, D. (2006), "Detecting spam web pages through content analysis", In *Proceedings of the 15th International Conference on World Wide Web*, *ACM*, pp. 83-92.

Patel, C.D. and Patel, J.M. (2017), "GUJSTER: a rule based stemmer using dictionary approach", In *Inventive Communication and Computational Technologies (ICICCT), 2017 International Conference on*, *IEEE*, pp. 496-499.

Quinlan, J.R. (1987), "Simplifying decision trees", *International Journal of Man-Machine Studies*, Vol. 27 No. 3, pp. 221-234.

Quinlan, J.R. (1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, CA, USA.

Ramya, R.S., Venugopal, K.R., Iyengar, S.S. and Patnaik, L.M. (2017), "Feature extraction and duplicate detection for text mining: a survey", *Global Journal of Computer Science and Technology*, Vol. 16 No. 5.

Rios, G. and Zha, H. (2004), "Exploring support vector machines and random forests for spam detection", *In CEAS*.

Sahami, M., Dumais, S., Heckerman, D. and Horvitz, E. (1998), "A bayesian approach to filtering junk e-mail", In Learning for Text Categorization: Papers from the 1998 Workshop, Vol. 62, pp. 98-105.

Sakkis, G. Androutsopoulos, I. Paliouras, G. Karkaletsis, V. Spyropoulos, C.D. and Stamatopoulos, P. (2001), "Stacking classifiers for anti-spam filtering of e-mail", arXiv preprint cs/0106040.

Schapire, R.E. (1990), "The strength of weak learnability", *Machine Learning*, Vol. 5 No. 2, pp. 197-227.

Sebastiani, F. (1999), "A tutorial on automated text categorisation", in *Proceedings of ASAI-99, 1st Argentinian Symposium on Artificial Intelligence*, *Buenos Aires, AR*, pp. 7-35.

Sergienko, R., Shan, M., and Schmitt, A. (2017), "A comparative study of text preprocessing techniques for natural language call routing", In Dialogues with Social Robots, Springer Singapore, pp. 23-37.

Shah, N.F. and Kumar, P. (2018), "A comparative analysis of various spam classifications", In Progress in Intelligent Computing Techniques: Theory, Practice, and Applications, Springer, Singapore, pp. 265-271.

Tang, Y., Krasser, S., He, Y., Yang, W. and Alperovitch, D. (2008), "Support vector machines and random forests modeling for spam senders behavior analysis", In *Global Telecommunications Conference, 2008, IEEE GLOBECOM 2008, IEEE*, *IEEE*, pp. 1-5.

Tripathi, A. and Trivedi, S.K. (2016), "Sentiment analyis of indian movie review with various feature selection techniques", In *Advances in Computer Applications (ICACA), IEEE International Conference on*, *IEEE*, pp. 181-185.

Trivedi, S.K. (2016d), "A study of machine learning classifiers for spam detection", In *Computational and Business Intelligence (ISCBI), 2016 4th International Symposium on*, *IEEE*, pp. 176-180.

Trivedi, S.K. and Dey, S. (2013a), "Effect of various kernels and feature selection methods on SVM performance for detecting email spams", *International Journal of Computer Applications*, Vol. 66 No. 21.

Trivedi, S.K. and Dey, S. (2013b), "Interplay between probabilistic classifiers and boosting algorithms for detecting complex unsolicited emails", *Journal of Advances in Computer Networks*, Vol. 1 No. 2.

Trivedi, S.K. and Dey, S. (2013c), "An enhanced genetic programming approach for detecting unsolicited emails", In *Proc. 2013 IEEE 16th International Conference on Computational Science and Engineering*, *Australia Published by IEEE Computer Society*, *Sydney*, 978-0-7695-5096-1/13 $31.00 © 2013 IEEE DOI 10.1109/CSE.2013.171

Trivedi, S.K. and Dey, S. (2013d), "Effect of feature selection methods on machine learning classifiers for detecting email spams", In *Proceedings of the 2013 Research in Adaptive and Convergent Systems*, *ACM*, (pp. 35-40).

Trivedi, S.K. and Dey, S. (2014), "Interaction between feature subset selection techniques and machine learning classifiers for detecting unsolicited emails", *ACM SIGAPP Applied Computing Review*, Vol. 14 No. 1, pp. 53-61.

Trivedi, S.K. and Dey, S. (2016a), "A novel committee selection mechanism for combining classifiers to detect unsolicited emails", *VINE Journal of Information and Knowledge Management Systems*, Vol. 46 No. 4, pp. 524-548.

Trivedi, S.K. and Dey, S. (2016b), "A comparative study of various supervised feature selection methods for spam classification", In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, *ACM*, (p. 64).

Trivedi, S.K. and Dey, S. (2016c), "A combining classifiers approach for detecting email spams", In *Advanced Information Networking and Applications Workshops (WAINA), 2016 30th International Conference on*, *IEEE*, pp. 355-360.

Witten, I.H. and Frank, E. (2005), *Data Mining: practical Machine Learning Tools and Techniques*, 2nd ed., Morgan Kaufmann series in data management systems, The United States of America.

Wu, S., Chen, Y.C., Li, X., Wu, A.C., You, J.J. and Zheng, W.S. (2016), "An enhanced deep feature representation for person re-identification", In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, *IEEE*, pp. 1-8.

Yang, Z., Nie, X., Xu, W. and Guo, J. (2006), "An approach to spam detection by naive bayes ensemble based on decision induction", In *Intelligent Systems Design and Applications, 2006. ISDA'06. Sixth International Conference on*, *IEEE*, Vol. 2, pp. 861-866.

Youn, S. and McLeod, D. (2007), "Efficient spam email filtering using adaptive ontology", In Information Technology, 2007. ITNG'07. Fourth International Conference on (pp. 249-254). *IEEE*.

Zhang, Y., Wang, S. and Wu, L. (2012), "Spam detection via feature selection and decision tree", *Advanced Science Letters*, Vol. 5 No. 2, pp. 726-730.

Zhang, Y., Wang, S., Phillips, P. and Ji, G. (2014), "Binary PSO with mutation operator for feature selection using decision tree applied to spam detection", *Knowledge-Based Systems*, Vol. 64, pp. 22-31.

Zhou, Z.H. (2012), *Ensemble Methods: Foundations and Algorithms*, Chapman and Hall/CRC.

Further reading

Aladdin Knowledge Systems (2018), "Anti-spam white paper", available at: http://www.eAladdin.com.

Duda, R.O., Hart, P.E., and Stork, D.G. (2012), *Pattern Classification*, John Wiley and Sons.

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., and Tibshirani, R. (2009), *The Elements of Statistical Learning*, New York, NY: Springer, Vol. 2, No. 1.

Schapire, R.E. (1997), "Using output codes to boost multiclass learning problems", In *ICML*, Vol. 97, pp. 313-321.

Corresponding author

Shrawan Kumar Trivedi can be contacted at: f10shrawank@iimidr.ac.in