Interplay between Probabilistic Classifiers and Boosting Algorithms for Detecting Complex Unsolicited Emails

Shrawan Kumar Trivedi and Shubhamoy Dey

***Abstract*—This paper presents the performance comparison of probabilistic classifiers with/without the help of various boosting algorithms, in the Email Spam classification domain. Our focus is on complex Emails, where most of the existing classifiers fail to identify unsolicited Emails. In this paper we consider two probabilistic algorithms i.e. "Bayesian" and "Naive Bayes" and three boosting algorithms i.e. "Bagging", "Boosting with Re-sampling" and "AdaBoost". Initially, the Probabilistic classifiers were tested on the "Enron Dataset" without Boosting and thereafter, with the help of Boosting algorithms. The Genetic Search Method was used for selecting the most informative 375 features out of 1359 features created at the outset. The results show that, in identifying complex Spam massages, "Bayesian classifier" performs better than "Naive Bayes" with or without boosting. Amongst boosting algorithms, 'Boosting with Resample' has brought significant performance improvement to the "Probabilistic classifiers".**

***Index Terms*—Unsolicited emails, probabilistic classifiers, boosting algorithms.**

I. Introduction

Email Spam or unsolicited bulk email is something that most of internet users continuously identify as a problem. It can be seen in the form of advertisements or some explicit content that may also carry some malicious code. These unsolicited Emails incur cost for a user, as these are unwanted as well as a time consuming process in terms of time required to sort them. Spam also consumes storage space, bandwidth, and processing time to maintain and transmit. A study reports that the volume of Spam is continuously increasing. Estimation shows that volume of spam was 7%, of the total global email traffic, in 2001, whereas in 2006, it ranged between 50% and 80% [1].

A large variety of Spam detection techniques such as blacklisting/whitelisting of the domain names, heuristics based filtering, and keyword based filtering have already been experimented with to handle this problem. However, the overall accuracy of these techniques is still a matter of concern. To overcome the disadvantages of the above techniques, Machine learning techniques have been introduced. By using machine learning, a Spam filter will be able to learn "What Spam is", as defined by the user. Therefore it can be expected that a well-trained filter can achieve a higher rate of accuracy. The process of assigning a given text massage to a particular category is known as Text Classification [2]. Some of the Machine learning classifiers occupy a prominent place in the Text Classification research.

Email Spam classification is now becoming critical due to the complexity involved in Spam massages. It is difficult to identify features that indicate many of the types of attacks by spammers. Some attacks, like Tokenisation (Splitting or modifying feature such as 'free' written as f r 3 3) and Obfuscation (hides feature from adding HTML or some other codes such as 'free' coded as fr&#101xe or FR3E), can change the information of a particular feature [3]. Several classifiers have been tested in such type of data but, most of them have not given accurate prediction. On the other hand, tests on probabilistic classifiers, such as Naive Bayesian [4], [5], and Bayesian classifier [6], [7], have demonstrated significant prediction accuracy. Bayesian classifiers, due to their interesting procedure to find most suitable words with help of deviation from the mean, are well-known in literature. In this paper, we will test the two Probabilistic Techniques mentioned above.

This paper presents the results of performance comparison of Probabilistic Classifiers with/without inclusion of Boosting Algorithms. We have considered three boosting algorithms: Bagging, Boosting with Resample and AdaBoost. These algorithms work as voting methods [8], which formulate a single classifier as a linear combination of a number of weak classifiers.

Genetic Search [9] method was used for searching of the most informative features. In literature, this search method has been shown to be effective. We then tested the classifiers on the features identified by this algorithm, which demonstrated the strength of this technique.

The later sections have been structured as follows: Section II focuses on the related work on the concern area. Section III describes the Boosting algorithms and their functioning. Section IV carries the description of the Probabilistic algorithms. Section V explains the working and strength of the Genetic search method. Section VI presents the Experiments and Evaluation. Section VIII presents the Analysis and at the last section concludes this work.

II. Related Work

A lot of work in the field classification has been reported. In Table I we have summarized the literature in the concerned area, in which classifiers have been tested on the different spam datasets.

Manuscript received January 21, 2013; revised March 25, 2013.

The authors are with Indian Institute of Management, Indore, India (email:f10shrawank@iimidr.ac.in).

DOI: 10.7763/JACN.2013.V1.27

TABLE I: LITERATURE REVIEW

Author(s) & Year	Model Used	Data Source / Data Set	Accuracy Achieved
Androutsopoulos et al. (2000) [10]	NB	Ling Spam corpus	83 to 99
Xavier Carreras and Lluis Marquez(2001) [11]	Boosting Tree	PU1 Corpus	97.12
Cormac O'Brien and Carl Vogel (2003) [12]	Bayesian Network	Ling-spam Corpus	76.9
Le Zhang, Jingbo Zhu, Tianshun Yao (2004) [13]	NB, ME, SVM, Boosting	PU1, Ling-spam, Spam Assassin, ZH1 Chinese Spam Corpus	NA
Xiao Luo, Nur Zincir-Heywood(2005) [14]	SOM, NB	Ling-Spam	96.7-99.0
Metsis et al. (2006) [15]	Five types of NB comparison	Enron data set with different compositions	90.5 to 96.6
Lai (2007) [16]	NB, KNN, SVM and SVM+Tf-Idf	C1_Total_16843 emails, C2_toal_24038_emails	82-91
Chen (2008) [17]	Bayesian classificatio n	PU1, PU2 corpus	92.8-96.2
Manjusha and Rakesh (2010) [18]	Bayesian and NN	Total_200_email, SpamRate_50%	98.5
Vahora et al. (2011) [19]	Vector Space Model with NB	Total_100_emails SpamRate_50%	85
Naveen Kumar Korada , et. al. (2012) [20]	NB + AdaBoost	Crop Data	33.3 improve

III. BOOSTING ALGORITHMS

The origin of the Boosting algorithms was in bootstrapping [21], [22]. Bootstrapping technique is basically used for assessing statistical accuracy of some estimate. It is a sample based statistical method which consists of drawing randomly with replacement from the set of data points. In the area of classification, some boosting algorithms have shown significant results.

A. Bagging

The bagging technique [21], [23], [24] takes the concept of Bootstrapping and uses it for classification purposes. It takes aggregation of the Bootstrap. Let us consider a training set $T = t_1, t_2, t_3 ... t_n$ with $t_i = (x_i, y_i)$. The main intension of this algorithm is to fit a regression model which will develop a prediction f_x at input x. By averaging prediction over the collection of Bootstraps, Bagging procedure will reduce variance and increase accuracy. We will fit our model by given prediction f_x^b for each Bootstrap Sample T^b with $b \in 1...B$. We can define the bagging estimation by-

$$f_x^{bag} = \frac{1}{B}\sum_{b=1}^{B} f_x^b \quad (1)$$

Algorithm for classification:

Input: Training set $T = t_1, t_2, t_3 ... t_n$ with $t_i = (x_i, y_i)$. Number of sample version of training set B.
Output: An appropriate classifier for the training set G_x.

For $n = 1...B$

1) Draw with replacement $K \le N$ sample from the Training set T, obtaining the n^{th} sample T^n.
2) Train Classifier G_n, for each sample T^n.
3) Build the Final Classifier as a vote of G_n with $n = 1...B$

$$G_x = sign(\theta_m \sum_{m=1}^{M} G_m^x) \quad (2)$$

B. Boosting with Re-sampling

Boosting with Re-sample technique is similar as Bootstrap and Bagging. The difference is, Bootstrapping and Bagging technique performs sampling with replacement but Boosting with Re-sample perform sampling without replacement. It was proposed in 1989 by Schapire [2].

Algorithm for classification:

Input: Training set $T = t_1, t_2, t_3 ... t_n$ with $t_i = (x_i, y_i)$. Number of sample version of training set B.
Output: An appropriate classifier for the training set G_x.

1) Draw without replacement $K^1 < N$ sample from the Training set T, obtaining the sample T^1.
2) Train weak Classifier G_1, for each sample T^1.
3) Select $K^2 < N$ sample from the training set T including half of the misclassified sample by G_1. Train weak classifier G_2 on it.
4) Select all remaining sample misclassified by G_1 and G_2. Train weak Classifier G_3 on it.
5) Build the final classifier based on the voting of weak classifier

$$G_x = sign(\sum_{n=1}^{3} G_n^x) \quad (3)$$

C. Adaptive Boosting (AdaBoost)

The idea behind Adaptive Boosting is to reweight the data instead of Random sampling. AdaBoost technique is a concept of building ensembles for the classifiers with improved performance. AdaBoost algorithm [24], [25], learns from the combination of output M of the weak classifier G_m^x, for the final decision of classification carried out by $G_x = sign(\theta_m \sum_{m=1}^{M} G_m^x)$.

Algorithm for classification:

Input: Training set $T = t_1, t_2, t_3 ... t_n$ with $t_i = (x_i, y_i)$. Number of sample version of training set B

Output: An appropriate classifier for the training set G_x.

- Initialise the weights $w_i^t = \frac{1}{N}$, $i \in \{1...N\}$
- For $m = \{1...M\}$.
  1) Learn the classifier G_m^x by the training data using the weights w_i^t.
  2) Calculate Error term $E_m^{rr} = \frac{\sum_{i=1}^{N} w_i^t I(y_i \neq G_m^{x_i})}{\sum_{i=1}^{N} w_i^t}$.
  3) Calculate weight contribution $\theta_m = 0.5 \log(\frac{1 - E_m^{rr}}{E_m^{rr}})$.
  4) Put $w_i^t \leftarrow w_i^t Exp(-\theta_m \mathrm{I}(y_i \neq G_m^{x_i}))$ then renormalize $\sum_i w_i^t = 1$.
- The final classifier

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

IV. Probabilistic Classifiers

This idea was proposed by Lewis [26], who developed the term $P(\frac{c^i}{d^j})$ which is known as probability of a document represented by a vector $d^j = \{w_1^j ... w_n^j\}$ of terms falls within a particular category c^i. This probability is calculated by the Bayes' theorem that can be seen as

$$P(\frac{c^i}{d^j}) = \frac{P(c^i) * P(\frac{d^j}{c^i})}{P(d^j)} \quad (5)$$

where $P(d^j)$ is probability of a randomly selected document represented as a vector d^j, and $P(c^i)$ is probability of the randomly selected document d^j falls on the class c^i. This is a basic method of the *Bayesian Filter*.

The equation of the Bayesian Classifier has been seen as problematic because of high number of the possible vectors d^j. This problem has been tackled by making the assumption that any two randomly selected coordinates of the document vector are statistically independent to each other. This assumption of independence can be determined by the equation 6.

$$P(\frac{d^j}{c^i}) = \prod_{l=1}^{n} P(\frac{w_l^j}{c^i}) \quad (6)$$

This assumption has been adopted by a classifier named *Naive Bayes* and it has been widely used in research related to the text mining [27]-[30].

V. Genetic Feature Search

In this paper, we have used Genetic search algorithm for the most informative feature selection. This algorithm is a type of inductive learning strategy that was initially introduced by Holland [31]. The algorithm works in a way similar to the Genetic models of the natural system, and that is why it is known as Genetic algorithm.

A Genetic algorithm initially maintains a constant population of individuals as sample of the space to be searched. Each individual is evaluated by its fitness. New individuals are formulated by choosing the best performing individuals who produce "offspring" [32] who retain the features of their parents. This creates a population with improved fitness.

Two main Genetic operators are involved in generating these new individuals. These operators are Crossover and Mutation. Crossover operator works by randomly selecting a point in two parents' gene structures and exchanging the remaining segments of parents to create new individuals. Therefore, crossover creates two new individuals by combining features of two old individuals. Mutation works by randomly changing some component of the particular individuals. It works as a population perturbation operator, which means adding new information in the population. Mutation operator also prevents any stagnation that might be occurring during the search process.

VI. Experiments and Evaluation

A. Data Set

We have taken our data from Enron email dataset. We selected Enron 4, Enron 5 and Enron 6 datasets and created 6000 HAM and 6000 SPAM files by random sampling. The reason for taking these versions of Enron emails is the complexities imbibed in Email SPAM. Complexity can be defined in terms of the attacks such as "Tokenisation", "Obfuscation" etc. done by spammers. Therefore, this dataset would demonstrate the efficacy of the classifiers against these attacks.

B. Pre-Processing of Data

The content of the Email files are represented by the vectors of the features i.e. weight of word i in document k. [33]. Then, these vectors are combined for a collection of documents to create a Term-Document Matrix. This process is called Indexing. Due to the large number of Email files, the resultant matrix would be very large and sparse. For this problem, some dimensionality reduction technique has to be used before classification. This can be done by "Feature Selection" or "Feature Extraction" methods. Dimensionality can be further reduced by "Stop word" removal (words that carry no information, such as "Pronouns, Prepositions and conjunctions") [33], Lemmatisation (grouping the words which have same meaning, such as "Boost, Boosted, Boosting").

C. Feature Selection

In this step, we select most informative features. Several techniques have been developed for feature selection. We have used the "Genetic Search" method. From the Term-Document matrix that carry 1359 attributes initially, the

Genetic Search Algorithm has selected 375 best features. These features were used for further analysis.

D. Classifiers

A Java and Matlab environment on window 7 operating system platform was used for the testing of the classifiers. Initially, we have evaluated some Probabilistic classifiers: "Bayesian" Classifier and "Naive Bayes" Classifier. Thereafter, Boosting algorithms were used for the purpose of boosting the performance of probabilistic classifiers. Three Boosting algorithms were tested: Bagging, Boosting with Re-sampling, and Adaptive Boosting (AdaBoost) and then compared with all combinations of these algorithms.

E. Evaluation

We have taken total 375 features for classifying 12000 Email files. The data was split into 66% training and 34% Test data. Concern classifiers will learn with the training data and remaining data will be taken for testing. We have taken accuracy (A_{cc}) and ($F_{H,S}$) value of the classifiers for our analysis.

The A_{cc} value is defined as

$$A_{cc} = \frac{S_c + H_c}{T_E} \tag{7}$$

where S_c is Total correctly classified Spam Text massage, H_c is Total correctly classified Ham Text massage and T_E is Total Text Email. The term accuracy can be defined as "The percentage ratio of total correctly classified Email to total Email". It shows the strength of a classifier i.e. what percentage of Emails is correctly classified.

The $F_{H,S}$ value is defined as

$$F_{H,S} = \frac{2 \times P_{H,S} \times R_{H,S}}{P_{H,S} + R_{H,S}} \tag{8}$$

where $P_{H,S}$ is the Precision and $R_{H,S}$ is the Recall. We have calculated both values separately for HAM and SPAM massages and then calculated the $F_{H,S}$ value. The final result will be calculated by the weighted average sum of the $F_{H,S}$ value.

By the help of these values, we will calculate the accuracy and strength of classifiers.

VII. Comparative Analysis

With the 375 best features selected by the Genetic search algorithm, Table II gives the significant performance comparison of the concern classifiers with and without boosting algorithms. It also shows the strength of genetic feature search method that has given up to 88.1% to 92.9% accurate classification of 12000 Email files (6000 HAM + 6000 SPAM).

If we compare the performance of probabilistic classifiers then in each case, "**Bayesian Classifier"** performance is better than Naive Bayes. It can be observed from Table II, Figure 1 and Figure 2, that Bayesian Classifier gives accuracy up to **88.7% to 92.9%** whereas, Naive Bayes gives, 88.1% to 91.7%.

TABLE II: Results of Classifiers with (out) Boosting

Boosting Algorithms	Probabilistic Classifiers	
	BayesNet (BN) Acc, F value In %	NavieBayes(NB) Acc, F value In %
Without Boosting	88.8, 88.7	88.0, 88.1
Bagging	89.2, 89.1	88.4, 88.4
Boosting with Re-Sample	**92.9, 92.9**	91.7, 91.7
AdaBoost	**92.4, 92.3**	91.2, 91.2

It also shows that without the use of boosting algorithms, both probabilistic classifiers give poor accuracy, which is 88.1% to 88.7%. With help of the boosting algorithms, the performance of these classifiers has increased.

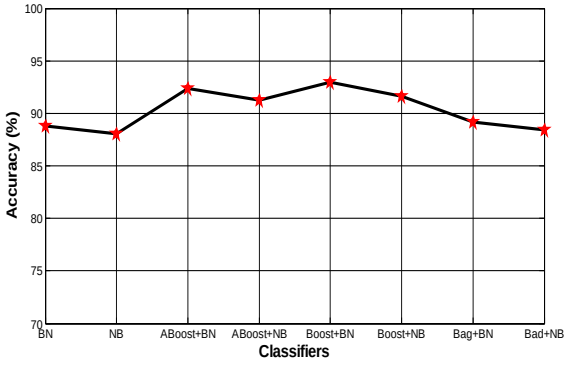


Fig. 1. Accuracy of Probabilistic Classifiers with(out) Boosting

For boosting of the performance of our probabilistic classifier, we have taken three Boosting algorithms. In the Table II and Fig. 1 and Fig. 2, it is clearly indicates that **Boosting with Resample** method gives maximum performance improvement. Table II, shows that the performance improvement with AdaBoost and Bagging is less than "**Boosting with Resample**". But the "**AdaBoost**" results are closer to best one. Finally, with Boosting done by "**Boosting with Resample", Bayesian Classifier** is giving best result. In this case, accuracy is up to **92.9%.**

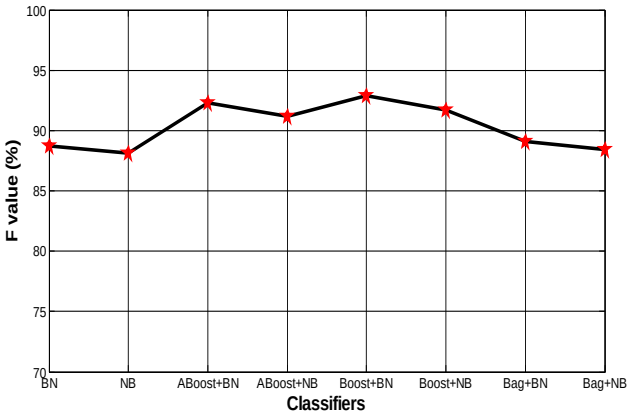


Fig. 2. F-Value of probabilistic classifiers with(out) boosting.

VIII. Conclusion

In this research, it has been shown that the Bayesian

classifier is a better predictor of the Spam than Naive Bayes. In this research, both the Accuracy and the F values have been used for gauging the strength of concerned classifiers. We obtained nearly similar values of these two measurements. We have also shown that Boosting algorithms play a crucial role in boosting the performance of classifiers. We have conducted three studies as a part of this research .to compare performance. First is the performance comparison of Probabilistic classifiers without Boosting where the Bayesian classifier has shown the best result. Second is the performance comparison of Boosting algorithms where Boosting with resample has shown significant strength, though the AdaBoost results were very close to the best one. And last is the performance comparison of Probabilistic Classifiers with Boosting where Bayesian Classifiers performed best when used in conjunction with boosting with resample.

As discussed in the previously boosting algorithms have significant contribution in boosting the performance of classifiers. As part of our future work, we will investigate the effect of boosting on the performance of other classifiers.

REFERENCES

[1] J. Zdziarski, *Ending spam: Bayesian content filtering and the art of statistical language classification*, San Francisco: No Starch Press, 2005.

[2] R. Schapire, "Using output codes to boost multiclass learning problems," in *Proc. 14th International Conf. on Machine Learning (ICML)*, Nashville, TN, 1997, pp. 313–321.

[3] J. Goodman, G. V. Cormack, and D. Heckerman, "Spam and the ongoing battle for the inbox," *Communications of the ACM*, vol. 50, issue 2, pp. 24-33, February 2007.

[4] T. Mitchell, *Machine Learning*, New York: McGraw-Hill, 1997.

[5] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proc. 17th Annu. International ACM-SIGIR Conf. on Research and Development in Information Retrieval*, 1994, pp. 3-12.

[6] D. Koller and M. Sahami, "Hierarchically classifying documents using very few words," in *Proc. 14th International Conf. on Machine Learning (ICML)*, Nashville, TN, 1997, pp. 170–178.

[7] P. Domingos and M. J. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss," *Mach. Learn*., vol. 29, no. 2-3, pp. 103–130, 1997.

[8] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants," *Machine Learning*, vol. 36, issue 1-2, pp. 105-139, 1999.

[9] J. H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: University of Michigan Press, 1975.

[10] Androutsopoulos, J. Koutsias, K. V. Chandrinos, G. Paliouras, and C. D. Spyropoulos, "An evaluation of naive bayesian anti-spam filtering" in *Proc. Workshop on Machine Learning in the New Information Age, 11th European Conf. on Machine Learning (ECML 2000)*, G. Potamias, V. Moustakis, and M. van Someren, Eds., Barcelona, Spain, 2000, pp. 9-17.

[11] X. Carreras and L. Màrquez, "Boosting trees for clause splitting," in *Proc. CoNLL-2001 Shared Task*, Toulouse, France, 2001.

[12] C. O'Brien and C. Vogel, "Spam filters: bayes vs. chi-squared; letters vs. words," in *Proc. International Symposium on Information and Communication Technologies*, M. Aleksy, *et al.*, Eds., 2003, pp. 298-303.

[13] L. Zhang, J. B. Zhu, and T. S. Yao, "An evaluation of statistical spam filtering techniques," *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 3, issue 4, pp. 243-269, December 2004.

[14] X. Luo and A. N. Zincir-Heywood, "Evaluation of two systems on multi-class multi-label document classification," *in Proc. Foundations of Intelligent Systems: 15th International Symposium, ISMIS (ISMIS 2005)*, Saratoga Springs, NY, USA, 2005.

[15] Metsis, I. Androutsopoulos, and G. Paliouras, "Spam filtering with naive bayes–Which naive bayes?" in *Proc. 3rd Conf. on Email and Anti-Spam (CEAS)*, 2006, pp.125–134.

[16] C. C. Lai, "An empirical study of three machine learning methods for spam filtering," *Knowledge-Based System*, Elsevier, vol. 20, issue 3, pp. 249-254, April 2007

[17] J. Chen and Z. Chen, "Extended Bayesian information criterion for model selection with large model spaces," *Biometrika*, vol. 95, no. 3, pp. 759-771, 2008.

[18] K. Manjusha and R. Kumar, "Spam mail classification using combined approach of bayesian and neural network," in *Proc. 2nd International Conf. on Computational Intelligence, Communication Systems and Networks (CICN'10)*, 2010, pp. 145-149.

[19] S. Vohra *et al.* "Novel approach: Naïve bayes with vector space model for spam classification," in *Proc. Nirma University International Conference on Engineering (NUiCONE), IEEE Conference*, 2011.

[20] N. K. Korada, N. S. P. Kumar, and Y. V. N. H. Deekshitulu, "Implementation of Naive Bayesian Classifier and Ada-Boost Algorithm Using Maize Expert System," *International Journal of Information Sciences and Techniques (IJIST)*, vol. 2, no. 3, pp. 63-75, 2012.

[21] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed., John Wiley & Sons, 2001.

[22] B. Efron, "The jack-knife, the bootstrap and other re-sampling plans," *CBMS-NSF Regional Conference Series in Applied Mathematics*, no. 38, 1982.

[23] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, issue 2, pp. 123–140, August 1996.

[24] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed., Springer, 2001.

[25] Y. Freund and R. Schapire, "Experiments with a new boosting algorithm," in *Proc. 13th International Conf. on Machine Learning (ICML)*, Bari, Italy, 1996, pp. 148–156.

[26] D. D. Lewis, "Naive (Bayes) at forty: The independence assumption in information retrieval," in *Proc. of 10th European Conference on Machine Learning (ECML-98)*, 1998, pp. 4 -15.

[27] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proc. 10th European Conference on Machine Learning (ECML-98)*, 1998, pp. 137-142.

[28] D. Koller, and M. Sahami, "Hierarchically classifying documents using very few words," in *Proc. 14th International Conf. on Machine Learning (ICML)*, D.H. Fisher, Ed., pp. 170-178, Morgan Kaufmann, San Francisco, 1997.

[29] L. S. Larkey and W. B. Croft, "Combining classifiers in text categorization," in *Proc. 19th Annual Conf. Research and Development in Information Retrieval (SIGIR-96)*, 1996, pp. 289-297.

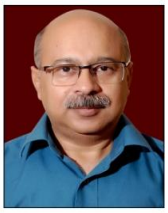
[30] A. Sharma and S. Dey, "A comparative study of feature selection and machine learning techniques for sentiment analysis," in *Proc. 2012 ACM Research in Applied Computation Symposium*, San Antonio, Texas, 2012.

[31] J. H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: University of Michigan Press, 1975.

[32] H. Vafaie and I. F. Imam, "Feature selection methods: Genetic algorithms vs. greedy-like search," in *Proc. 3rd International Fuzzy Systems and Intelligent Control Conference*, 1994.

[33] K. Aas and L. Eikvil, "Text categorisation: A survey," technical report, Norwegian Computing Centre, June 1999.

Shrawan Kumar Trivedi is a research fellow of information systems at Indian Institute of Management Indore, India. He completed his Master of Technology in Information Technology from Indian Institute of Information Technology (IIIT-Allahabad) and Master of Science in Electronics and Communication from UIET, CSJM University Kanpur, India. Currently he is working on Data Mining.



Shubhamoy Dey is an associate professor of information systems at Indian Institute of Management Indore, India. He completed his Ph. D from the School of Computing, University of Leeds, UK, and Master of Technology from Indian Institute of Technology (IIT- Kharagpur). He specializes in Data Mining and has 25 years of research, consulting and teaching experience in UK, USA and India.