Journal of King Saud University – Computer and Information Sciences 34 (2022) 2733–2742

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

A novel improved random forest for text classification using feature ranking and optimal number of trees

Nasir Jalal [a], Arif Mehmood [a], Gyu Sang Choi [b,*], Imran Ashraf [b,*]

[a] Department of Computer Science and Information Technology, The Islamia University of Bahawalpur, Bahawalpur 63100, Pakistan
[b] Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Republic of Korea

ARTICLE INFO

Article history:
Received 14 November 2021
Revised 10 March 2022
Accepted 10 March 2022
Available online 31 March 2022

Keywords:
Improved random forest
Text classification
Feature ranking
Decision tree optimization
Machine learning
Feature reduction

ABSTRACT

Machine learning-based models like random forest (RF) have been widely deployed in diverse domains such as image processing, health care, and text processing, etc. during the past few years. The RF is a prominent technique for handling imbalanced data and performs significantly better than other machine learning models due to its parallel architecture. This study presents an improved random forest for text classification, called improved random forest for text classification (IRFTC), that incorporates bootstrapping and random subspace methods simultaneously. The IRFTC removes unimportant (less important) features, adds a number of trees in the forest on each iteration, and monitors the classification performance of RF. Classification accuracy is determined with respect to the number of trees which defines the optimal number of trees for IRFTC. Feature ranking is determined using the quality of the split in a tree. The proposed IRFTC is applied on four different benchmark datasets, binary and multiclass, to validate its performance in this study. Results indicate that IRFTC outperforms both the traditional RF, as well as, other machine learning models such as logistic regression, support vector machine, Naive Bayes, and decision trees.

1. Introduction

Text classification has gained ample interest during the last decade due to the ever-increasing size of generated data online. With petabytes of data from social media and other platforms, the classification and analysis of such data can provide plentiful information about the public interest, opinions, and trends. With promises of valuable insights on the data, text classification poses several challenges due to text veracity, high dimensionality, unbalanced distribution, etc. (Xu et al., 2012). In response, a large number of text classification models such as logistic regression (LR) (Hosmer et al., 2013), support vector machine (SVM) (Noble, 2006), Naive Bayes (NB) (Murphy, 2006) and their extended variants such as ensemble models have been introduced (Mehmood et al., 2017; Khalid et al., 2020; Umer et al., 2021).

Random forest (RF) is one of the best classifiers widely used for regression and classification tasks. Algorithmic simplicity makes it an attractive choice for text classification. In addition, its capability to handle the high dimensional data and high performance under imbalanced datasets are significant advantages over other machine learning models (Luo et al., 2015; Paul and Mukherjee, 2015; Khoshgoftaar et al., 2007; Liu et al., 2008; Dietterich, 2000). Based on 'wisdom of the crowd', RF utilizes a large number of decision trees for decision making. To make the final decision, it uses the average or mean of decision trees outputs, thereby providing more accurate results as compared to the decision tree. Besides text classification, currently, RF is widely used in multitudes of domains including image processing, data analysis in the health care sector, agriculture sector, crime detection, etc. (Ashraf et al., 2018; Criminisi and Shotton, 2013). RF is used for handling non-linear classification tasks.

Together with the traditional RF, several variants are have also been presented (Quadrianto and Ghahramani, 2014) that enhance its classification performance with the increase of the number of trees (Breiman, 2001). However, experiments show that the performance cannot be improved further with the addition of trees beyond certain limits (Oshiro et al., 2012). In this regard, studies (Latinne et al., 2001; Cuzzocrea et al., 2013) worked on finding the optimal trees to achieve the best performance of RF. Improved random forest (IRF) has been introduced that adds methods for feature selection and an optimal number of trees at the same time. The IRF starts with a small number of trees and finds a diverse number of features. Then at each stage, important features are selected and unimportant features are discarded using a feature

* Corresponding authors.

https://doi.org/10.1016/j.jksuci.2022.03.012
1319-1578/

importance criterion. Moreover, the IRF distributes the features in different categories such as important and unimportant features. Based on these features, the limit of the upper bound in the number of trees to be added in the forest at each phase can be evaluated. The working of IRF on the improved random forest for image processing application can be found in Paul et al. (2018). This study makes the following key contributions.

- This study extends the application of Paul et al. (2018) for text classification and names it improved random forest for text classification (IRFTC).
- In the proposed IRFTC, the number of trees is not predefined like previous methods, therefore, it is not data-dependent. IRFTC is fast and useful for many types of industrial applications.
- Contrary to traditional RF, the proposed IRFTC follows an iterative process regarding feature reduction. On each iteration, the most insignificant features are automatically removed.
- IRFTC uses an optimal number of trees while traditional RF infers that increasing the number of trees automatically increases the accuracy which is not very practical.
- The proposed IRFTC improves the performance by considering two factors: features and tree optimality. It removes unimportant (less important) features, adds a number of trees in the forest on each iteration, and monitors the classification performance of RF.
- Experimental results show that the addition of a number of trees satisfies the upper bound, as well as, improves the classification accuracy. Consequently, the proposed IRFTC provides optimal classification accuracy in terms of the addition of trees and reduction of features.

The rest of the paper is structured as follows. Section 2 discusses several important research works related to the current study. The proposed IRFTC model and its related mechanisms are described in Section 3. Section 4 provides the discussion of results while, in the end, the conclusion is given in Section 5.

2. Related work

Keeping in view the large amount of data generated on the internet, text classification has emerged as an important research area. The RF shows better performance for text classification (Xu et al., 2012; Paul et al., 2018). It can cope with text-related challenges such as noise, sparsity, data imbalance, etc. (Chaudhary et al., 2016; Jacob, 2015). Consequently, several modifications and extensions of RF have been proposed (Paing and Choomchuay, 2018; Kalaiselvi and Thangamani, 2020).

For example, Paul et al. (2018) proposed a method for detection of mitosis from different types of historiographical images. Furthermore, it can perform histopathological grading of cancer that helps in making individual treatment plans. Mitosis calculation plays an important role in breast cancer grading using the grading system. Traditionally, such grading is performed by pathologists using manual examinations of thousands of images for each patient. So, finding the mitotic figures automatically is an important, yet challenging task. Study results show that the segmented cells can be classified in mitotic and non-mitotic using RF. Simulation results show that a 12% improvement in the F1 score can be obtained using the proposed RF classifier. Similarly, Quadrianto and Ghahramani (2014) proposed a new approach to generate an RF that performs random sampling of many trees before distribution and combines the predictive probabilities. The proposed approach performs the sampling of decision trees even before looking at the data. Due to this, each tree performs prediction with the use of the Bayesian method to make Bayesian RF with security. Results show that the proposed method gives outstanding results in terms of accuracy and speed. Study Nugroho et al. (2019) uses RF to detect hate speech and offensive words and compares its performance with AdaBoost and neural network (NN). Results show that RF has a better classification accuracy as compared to AdaBoost and NN.

Feature selection is a fundamental part of RF for improving its performance, and (Kou et al., 2020) investigates feature selection methods by using multiple criteria decision making (MCDM) methods in that regards as the single measure is not enough to evaluate features. Evaluations of feature selection are carried out based on performance, stability, and efficiency. To solve the features conflict, the study leverages the MCDM to resolve it. Empirical results show that multiple criteria are necessary for the evaluation of feature selection. To overcome the process of feature selection and an optimal number of trees, (Paul et al., 2018) introduces the IRF that achieves feature reduction and addition of trees simultaneously. IRF algorithm provides optimal classification accuracy concerning the number of terms and feature reductions. The study does not predefine the number of trees in the forest. It finds local and global weights of features with the help of out-of-bag estimates. After the calculation of the weight, the IRF finds important and unimportant features on the basis of global weights. Results show that IRF removes unimportant features, as well as, controls the forest size dynamically, and gives optimum performance dynamically. Along the same lines, Feng et al. (2020) proposes feature selection-based RF (FSRF) to improve random forest for text classification. To avoid the overfitting problem of RF caused by fixing the number of the decision tree, FSRF uses feature selection to obtain the best performance. FSRF uses three methods of feature selection that is Filter, embedded, and wrapper method for the selection of features. Only those features are considered that increase the classification performance to generate a feature subset. A sparse matrix projection is also introduced to improve the process of random forest. Experimental results indicate that FSRF reduces the complexity of the model, avoids overfitting, and improves classification accuracy.

The study uses an improved RF for the classification of big data (Lakshmanaprabu et al., 2019). For selecting the optimal features, an improved dragonfly algorithm (IDA) algorithm is used. The size reduction is performed using the MapReduce framework. Experimental results show that the performance of the proposed RF is high as compared to traditional RF with respect to the accuracy, precision, and recall. Similarly, news text classification is investigated using RF and NB with TFIDF features (Parida et al., 2021). Results prove the superiority of RF over NB for news categorization. An improved RF classifier for text categorization is presented in Xu et al. (2012). Although, it also uses the random subspace method and tree optimization but the correlation is performed between feature value and distinct label value. On the other hand, the current study finds the correlation between pair of trees with the hypothesis that heterogeneous trees in the forest give better accuracy. Further, the tree selection in Xu et al. (2012) is biased in the sense that it combines top 70% trees with high out of bag accuracy in improved RF.

3. Materials and methods

The IRFTC model proposed in this study is inspired by Paul et al. (2018) that applied RF to segment the images. The proposed model IRFTC is an iterative process. At each iteration, the most insignificant features are removed automatically. The proposed model contains preprocessing, feature ranking, discriminating important and unimportant features, and finding the optimal number of trees. These modules are discussed separately in the following sections.

3.1. Preprocessing

Preprocessing of IRFTC consist of three steps. First, it takes the dataset as a text file as shown in Table 1. Secondly, the dataset is preprocessed involving tokenization, case conversion, punctuation removal, number removal, stopwords removal, and stemming. These steps are performed to remove unnecessary words and symbols that do not contribute to the learning process. Preprocessed data is later used for feature extraction for models' training. This study used the term frequency-inverse document frequency (TFIDF) as the feature extraction approach.

To show the influence of preprocessing, two samples from the raw data are used for TFIDF extraction before and after preprocessing. TFIDF is calculated using the following formulas for TF and IDF

$$TF = \frac{t_i}{d_i} \tag{1}$$

where t_i is a unique term while d_i shows the number of terms in a document d.

$$IDF = Log(\frac{D}{t_i}) \tag{2}$$

In the end, TF-IDF feature vector is obtained by multiplying the TF and IDF as follows

$$TFIDF = TF \times IDF \tag{3}$$

The text used for this example is, 'I see the letter B on my car' for 'ham' class, while for 'spam' class, 'To get 2.50 lb free call credit' is used. Initially, TF is calculated for both sample texts and is shown in Table 2.

Similarly, the calculated IDF for the sample text is given in Table 3.

In the end, TF and IDF are multiplied to obtain the TFIDF features. Table 4 shows the extracted TFIDF without the preprocessing steps while the results after preprocessing are given in Table 5.

3.2. IRFTC algorithm

We introduce IRFTC to find the optimal number of trees and reduce features for which strength is saturated. So increase in the number of trees in the forest aims at further increasing the correlation, not accuracy. For convenience, Table 6 describes the symbols used in this paper.

Feature reduction and the number of trees addition are performed simultaneously. IRFTC chooses the features for the forest from both important and unimportant bags of features which reduces the chance of ignoring important features. Each decision tree in our IRFTC is built using the bootstrap sample. Let, F be the number of features in the bootstrap sample, then f features are selected for construction of decision trees where $f < F$.

For each node split, a subset of features f is selected from F. only one feature is selected for node splitting from the λ set of features. In IRFTC, trees are added recursively. Study Paul et al. (2018) observed that not all the feature has the same weight, so features are divided into two groups: important and unimportant features. The process starts with the base number of trees in a random forest. For each iteration, the random forest is updated with new important and unimportant features. The following five core steps are involved in our IRFTC algorithm. First, we assign weights to each feature. Second, important and unimportant features are separated based on the threshold value. Third, the features are ranked, followed by the process of finding the maximum number of trees in the fourth step. In the end, the forest is converged and optimal results are taken from IRFTC. For each iteration, the same strategy is applied and the process of adding trees in the forest is repeated until optimal classification is achieved. The overall mechanism of IRFTC is depicted in Fig. 1.

Table 1
Sample raw text from the dataset.

Text	Class
Nah I don't think he goes tousf, he lives around here though	Ham
I'm gonna be home soon and i don't want totalk about this stuff anymore tonight, k? I 've cried enough today.	Ham
URGENT! You have won a 1 week FREE membership in our $100,000 Prize Jackpot! Txt the word: CLAIM to No. 81010 T&C WIN!	Spam
WINNER!! As a valued network customer you have been selected to receive $900 prize reward! To claim call 090617014.	Spam

Algorithm 1: IRFTC algorithm.

1: Initialize random forest Φ_o, random trees T_o and feature vector F_o
2: Find global weight using H and rank them using (4)
3: Sort feature and put top $I_o = \sqrt{F_o}$ in BI
4: $U_o = F_o - I_o$ put U_o in BU
5: Initialize n = 0
6: **while** $U_n > f$ **do**
7: Compute μ_n and σ_n from BU_n
8: Calculate R_n from BU_n
9: From BU_n, find $H(j)$ where $H(j) >$ min (BI_n) 10: $F_{n+1} = F_n - R_n$
11: $BI_{n+1} = B_n + A_n$, $BU_{n+1} = BU_n - R_n - A_n$
12: $I_{n+1} = BI_{n+1}$
$U_{n+1} = BU_{n+1}$
$\Delta I = I_{n+1} - I_n$
$\Delta U = U_{n+1} - U_n$
13: From Eq. (26) ΔT, $T_{n+1} = T_n + \Delta T$
14: Grow ϕ_{n+1} with Tn + 1 number of trees and features vector F_{n+1}
15: Find H and rank all features in F_{n+1}
16: Increment in n
17: **end while**

The sequence of steps for IRFTC is provided in Algorithm 1. We move with initial random forest ϕ_0 which have T_o number of trees. The F_o is the initial set of the feature vector. After feature ranking we mark top $I_o = \sqrt{F_o}$ features as important features. At this stage, we have two bags of feature B_o and $B\prime_o$. In B_o we put important features while the rest of the features (unimportant features) in $B\prime_o$. Let μ_o and σ_o be the mean and standard deviation of weights present in $B\prime_o$, then from $B\prime_o$ we calculate $\mu_o - 2\sigma_o$ and remove all such features whose global wights are less than $\mu_o - 2\sigma_o$ and put into new Bag R_o. After removing feature from $B\prime_o$, we update our feature vector i.e. $F_1 = F_o - R_o$. After updating feature vector, we observe change in U and I. Based on ΔU , $\Delta I, P_U, P_I$ and l we calculate ΔT and update our forest Φ_1. The process of updating feature vector and forest consists of n number of steps/iterations. Final step will be $F_{n+1} = F_n - R_n$, similarly our final forest Φ_n grows with $T_n + \Delta T$ number of trees and F_{n+1} feature vectors. The processes for feature ranking, the optimal number of trees, and an ensemble of iteration are given in the following sections.

3.3. Feature ranking for text classification

Let F be all the features and f be the set of selected features. Feature ranking is performed with the help of split quality (QS) on each node (κ) of the tree(Υ), the weight of feature (H), and out of bag error estimate (ε). The whole ranking process is depicted in Fig. 2.

Table 2
Term frequency of unprocessed sample text.

car	call	credit	B	free	To	2.50	pounds	on	my	letter	see	get	the	I
0.125	0.000	0.000	0.125	0.000	0.000	0.000	0.000	0.125	0.125	0.125	0.125	0.000	0.125	0.125
0.000	0.143	0.143	0.000	0.143	0.143	0.143	0.143	0.000	0.000	0.000	0.000	0.143	0.000	0.000

Table 3
Inverse document frequency of unprocessed sample text.

car	call	credit	B	free	To	2.50	pounds	on	my	letter	see	get	the	I
0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301
0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301	0.301

Table 4
TFIDF from the raw text without preprocessing.

car	call	credit	B	free	To	2.50	pounds	on	my	letter	see	get	the	I
0.038	0.000	0.000	0.038	0.000	0.000	0.000	0.000	0.038	0.038	0.038	0.038	0.000	0.038	0.038
0.000	0.043	0.043	0.000	0.043	0.043	0.043	0.043	0.000	0.000	0.000	0.000	0.043	0.000	0.000

Table 5
TFIDF from the raw text after preprocessing.

call	pounds	letter	free	see	car	credit	get
0.000	0.000	0.038	0.000	0.038	0.038	0.000	0.000
0.043	0.043	0.000	0.043	0.000	0.000	0.043	0.043

Suppose we have data set with F_o initial feature vector during the development of the decision tree. In the decision tree, we use entropy-based node splitting. We denote node κ of decision tree Υ. The probability of class label l at node K is denoted by $P(l)$. We define the entropy of that node κ as

$$E = \sum P(l) log \frac{1}{p(l)} \tag{4}$$

Table 6
Symbols and their Description.

Symbol	Description
QS	Quality of Split
κ	Single Node in a Tree
K	All Nodes in a Tree
Υ	Single Tree
T	All Trees in Forest
H	Weight of Tree
ε	Out of Bag Error
I	Number of Important Features
U	Number of Unimportant Features
BI	Bag of Important Features
BU	Bag of Unimportant Features
$\mu - 2\sigma$	Threshold Value
P	Probability of Good Split
Q	Probability of Not Good Split
F	All Features in Dataset
F_o	Initial Feature Vector
F_n	Feature Vector after nth iteration
f	Sample of Features
λ	Set of features (Both Important and Unimportant)
C_f	Correlation of Forest
ϕ	Forest
S_f	Forest Strength
A	Classification Accuracy of Forest

For splitting the node κ we select f feature without replacement from initial feature vector F_o. Let node κ be split by feature j, the entropy of left and the right node is denoted by E_l and E_r, respectively, then the quality of split at node κ by feature j can be defined

$$H^{\Upsilon}(j) = \frac{\sum_{\kappa=1}^{K} QS(\kappa, j)}{K} \tag{5}$$

We select feature j that provides the highest quality of split QS. Now we calculate local weights for feature j of tree Υ. Eq. (5) the highest value of $H^{\Upsilon}(j)$ shows the better quality of split of feature j in tree Υ but we have to calculate the weights of all trees using out of bag error (ε).

Let ε^{Υ} be the out of bag error for tree Υ, now on the behalf of out of bag error normalize weight of tree Υ is calculated. We try to reduce out of bag error by calculating the normalized weight of trees

$$H^{\Upsilon} = \frac{1/\varepsilon^{\Upsilon}}{MAX_{\Upsilon}(1/\varepsilon^{\Upsilon})} \tag{6}$$

The relationship between classification error and H^{Υ} is inverse as high value of H^{Υ} indicates low classification error of tree κ using Eqs. (5) and (6) we find global weight of feature j as

$$H(j) = \frac{\sum_{\Upsilon} H^{\Upsilon}(j)(H)^{\Upsilon}}{MAX_j \sum_{\Upsilon} H^{\Upsilon}(j)(H)^{\Upsilon}} \tag{7}$$

The high value of $H(j)$ indicates the importance of feature j. Using the global weight of feature j, we rank all the features which help to discriminate between important and unimportant features. We develop a state-of-the-art strategy to distinguish between important and unimportant features. From the ranked list of

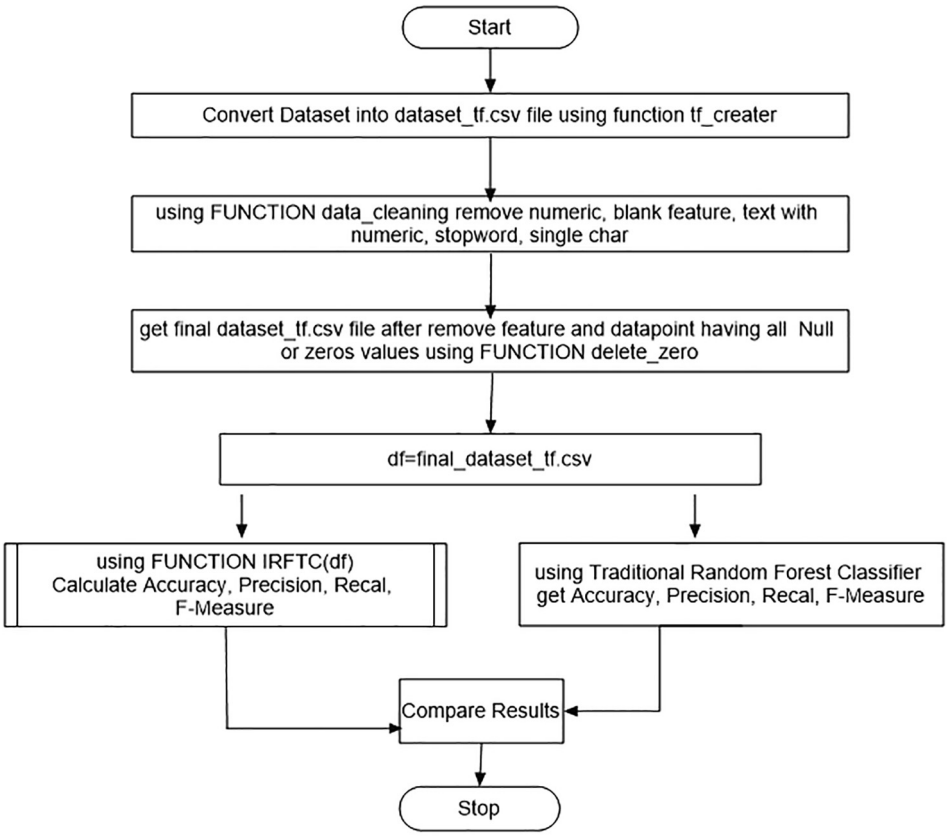

Fig. 1. The sequence of operations followed in IRFTC.

features, we select the top I features as important and the rest of the features as unimportant. After discrimination of important and unimportant, we put important feature BI and unimportant feature in BU. We fix a threshold value $\mu - 2\sigma$ where μ and σ are the mean and standard deviation of BU, respectively. From BU the feature that has a weight less than $\mu - 2\sigma$ will be removed from BU and put in R where BU is updated at every iteration. We remove the feature present in R. If the global weight $H(j)$ of an unimportant feature is greater than the minimum weight of important features, then we put such features into the bag of important features which reduces further ignoring important features.

$$H(j) >= j^{MIN} H(j') \tag{8}$$

Now we update the bag of important and unimportant features with the addition of a new feature in BI and removal of a feature from BU. On each construction pass, we update our bags so we observe the change in BU and BI.

$$\Delta U = (BU_{i+1}) - (BU_i) \tag{9}$$
$$\Delta I = (BI_{i+1}) - (BI_i) \tag{10}$$

Using $U, I, \Delta U$, and ΔI we find how many trees will be added in IRFTC.

3.4. Finding optimal number of trees

For finding the optimal number of trees in the forest, it is assumed that the average number of nodes in a tree remains the same if we increase the number of trees in the forest. We want to improve the classification accuracy of random forest. As Paul et al. (2018) pointed out that the classification accuracy depends upon multiple factors including strength and correlation of trees while strength and correlation depend on the probability of good split, as shown in Fig. 3.

3.4.1. Good split

If a node is split by an important feature, then there is a possibility of a good split. A good split occurs if at least one important feature is present in a set of features from which we select features randomly for node splitting. As said earlier, the performance (classification accuracy) of RF depends on the strength and correlation of trees, and strength and correlation depend upon the probability of a good split.

Let P be the probability that at least one important feature is present for node splitting in a tree, $q = \neq gp = 1 - p$ is the probability that in all nodes none of the split is done by an important feature. A split is said to be good if a split is done by an important feature. If P is the probability of good split at a single node (κ) then P^K is the probability of good split for all nodes K in tree Υ. A good split creates a child node homogeneous within a tree. Now we find the probability p that at least one important feature is present in the set of features, where q is the probability that no important feature is present in a set of features i.e $q = 1 - p$

$$q = 1 - p = \frac{\binom{U}{f}}{\binom{U+I}{f}} \tag{11}$$

If $U < f$, it means that at least one important feature is always present in a set of features, hence one important feature must be selected. So when $u < f$ we have $p = 1$ and $q = 0$. Let our forest consists of T number of trees denoted by $P_I = \frac{\partial p}{\partial I}$ and $P_U = \frac{\partial p}{\partial U}$.

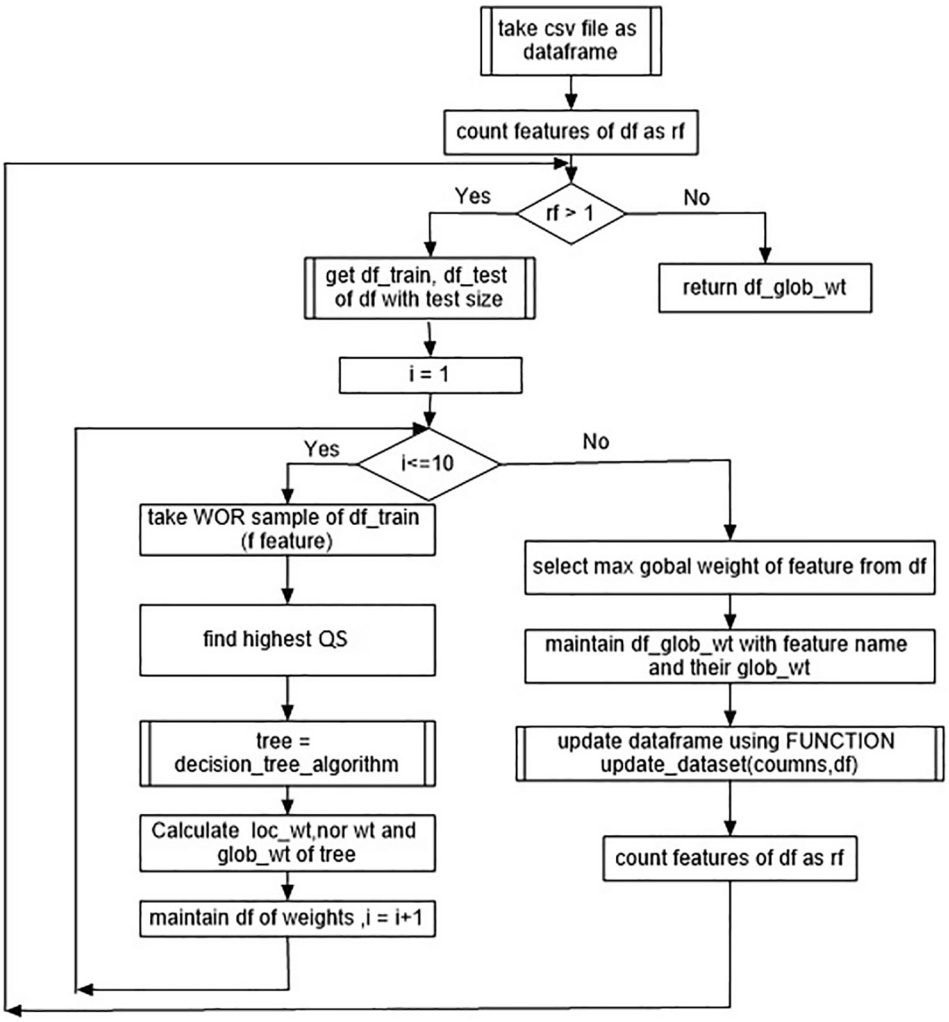

Fig. 2. The module used for feature ranking in IRFTC.

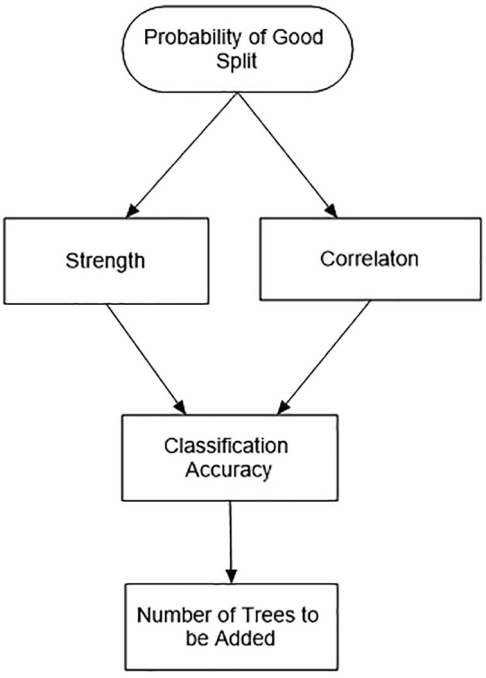

Fig. 3. Criteria for number of trees to be added.

After taking the partial derivative of Eq. 11, we get $P_I = -\frac{\partial q}{\partial I}$ and $P_U = -\frac{\partial q}{\partial U}$.

We find the approximate value of $-\frac{\partial q}{\partial U}$ by taking the partial derivative of q with respect to I and U,

$$P_I = -(\frac{\Delta q}{\Delta I})_{T,u} \tag{12}$$

$$P_I = \frac{I!(I+U-1-f)!f}{(I-f)!(I+U)!} \tag{13}$$

$$P_U = -(\frac{\Delta q}{\Delta U})_{T,I} \tag{14}$$

$$P_U = -\frac{(U-1)!(I+U-1-f)!If}{(U-f)!(I+U-1)!(I+U)} \tag{15}$$

From Eqs. 11 and 15, we see that $PI > 0$ and $PU < 0$. Now with the help of P we calculate strength and correlation. Before calculating strength and correlation we find the average number of nodes per tree. As per Paul et al. (2018), the average number of nodes per tree is independent of the number of trees in the forest.

3.4.2. Strength

The strength of a forest is defined as the minimum accuracy of an individual tree (Oshiro et al., 2012).

Let N_{avg} be the average number of nodes per tree. As the calculating probability of good split at a single node is p, for N_{avg} number

of nodes probability of good split is P. Classification accuracy is proportional to PN_{avg}. Since our forest consists of T number of trees we use the binomial distribution to find the strength of the forest. Strength of forest (S_f) is the probability of all nodes in the forest with at least one tree having a good split as given in Eq. 16.

$$S_f = 1 - (1 - P^{Navg})^T \tag{16}$$

After calculating the strength of the forest, we find the correlation between trees.

3.4.3. Correlation

As discussed earlier, the nodes within trees should be homogeneous and trees in a forest should be as heterogeneous as possible. If our forest consists of T number of trees, $T/2$ pair of trees exist in the forest. Let Υ_1 and Υ_2 be the pair of trees built from two different set of features λ_1 and λ_2, respectively. We can find the probability p that at least one feature is common from both λ_1 and λ_2 set of features.

$$\rho = 1 - \frac{\binom{I+U-f}{f}}{\binom{I+U}{f}} \tag{17}$$

For N_{avg} pair of nodes in tree Υ_1 and Υ_2 at least one feature is common.

$$\rho = (\rho')^N = (1 - \frac{\binom{I+U-f}{f}}{\binom{I+U}{f}})^N \tag{18}$$

For correlation of forest

$$C_f = \sum_{\Upsilon=1}^{T} \binom{T/2}{\Upsilon} \rho^{\Upsilon} (1-\rho)^{T/2-\Upsilon} \tag{19}$$

Based on the strength and correlation, we find the accuracy of forest A

3.4.4. Classification accuracy vs number of trees

Classification accuracy of the forest is based on strength and correlation.

$A = \lambda(S_f - C_f)$

From Eqs. 16 and 19 we get

$$A = \lambda((1-\rho)^T - (1 - p^{Navg})^T) \tag{20}$$

We calculate l, p, from U, and I, so it is clear from Eq. 20 that A is function of I, U and T.

$$d_A = \lambda[\frac{\delta(S_f - C_f)dT}{\delta T} + \frac{\delta(S_f - C_f)dI}{\delta I} + \frac{\delta(S_f - C_f)dU}{\delta U}] \tag{21}$$

$$U = \frac{(1-\rho)^{T/2}}{2} \log(1-\rho) - (1 - P^{Navg})^T \log(1 - P^{Navg}) \tag{22}$$

From Eq. 20 we get

$$\frac{\delta(S_f - C_f)}{\delta U} = -\frac{T}{2}(1-\rho)^{T/2-1}\frac{\delta P}{\delta U} + TNavgP^{Navg-1}(1 - P^{Navg})^{T-1}\frac{\delta P}{\delta U} \tag{23}$$

Let $l = TN_{avg}$, then

$$\frac{\delta(S_f - C_f)}{\delta I} = \rho\frac{\delta P}{\delta I}\frac{\delta(S_f - C_f)}{\delta U} = \rho\frac{\delta P}{\delta U} \tag{24}$$

$$d_A = \lambda(U \triangle T + lP_U \triangle U + lP_I \triangle I) \tag{25}$$

The d_A must be positive to improve the classification accuracy. So to improve the classification $P_I > 0$, $P_u < 0$ and $l > 0$ and $\Delta U < 0$. $\Delta U < 0$ supports our hypothesis that unimportant features and classification are inversely proportional. So to ensure $d_A > 0$ from Eq. 22, we have

$$|U\Delta T| < |\rho(P_I\Delta I + P_U\Delta U)||\Delta T| < |\rho\frac{(P_I\Delta I + P_U\Delta U)}{U}| \tag{26}$$

To find ΔT, we have to find P_I and P_U. According to Paul et al. (2018), it is observed that adding more trees in the forest surely increases the classification accuracy so based on the above argument we take the n iteration ensemble to obtain the best results.

3.5. Ensemble of iteration

Our vital target is to reduce U such that $(U < f)$ as a result P increase to 1 consequently.

$P_i = \frac{\delta P}{\delta I} = 0$and

$P_u = \frac{\delta P}{\delta u} = 0$

From Eqs. 25 and 26, we get.

$d_A \simeq \lambda(\frac{(1-\rho)^{T/2}}{2}\log(1-p))\Delta T$ At this point $S_f = 1$ and $A = 1$, So when $u < f$ classification accuracy get its peak and further increase in T only increases the correlation not accuracy. Obviously when $u < f$ we cannot increase A, and we use our algorithm for testing. As discussed earlier, $u < f$ classification accuracy is its peak so from line 10 of Algorithm 1 BU will be updated on every construction pass to reduce BU. For every pass of the algorithm, U is definitely removed so when $u < f$, we make an ensemble of the forest. After finding the number of trees to be added we converge our IRFTC algorithm and compare the result of IRFTC with traditional random forest.

3.6. Testing of IRFTC algorithm

The ensemble of forests obtained from the IRFTC algorithm is helpful for obtaining better classification performance for benchmark datasets. We apply test data at the root node and get the class label from the leaf node. Let ensemble forest consist of T^* number of trees and H^* be the weight of tree Υ in the forest, P^* be the probability of class label predicting by tree Υ then actual class label C^* can be obtained using

$$C^* = argmax_c \sum_{\Upsilon=1}^{T} H^*(\Upsilon)P^*(c) \tag{27}$$

4. Results and discussions

4.1. Experiment setup and datasets

This study conducts experiments using an Intel Core i7 11th generation system operating on Windows 8.1 64 bit. The algorithms are implemented using the Python programming language and Scikit-learn library 0.22.2 post1 and sklearn 10.0.6 are used for machine learning models. To analyze the performance of the proposed IRFTC, several well-known machine learning algorithms are implemented including decision tree (DT), RF, logistic regression (LR), support vector machine (SVM), and Naive Bayes (NB).

DT is a simple, yet popular machine learning algorithm and widely used to infer decisions based on the relationship between features. Each tree has a root node, internal node, and leaf node and each node represents a feature. The binary tree is the most common form of DT used in classification tasks (Loh, 2011). The tree is divided using a split criterion and the gain ratio is the most common split criterion. DT is non-parametric, computationally

inexpensive, and easy to interpret. Also, feature redundancy does not influence the performance of a DT.

RF is a tree-based ensemble model that uses decision trees built on the data attributes. Each tree is generated using a random vector from the input. Based on the ensemble architecture, each tree casts one vote for the final prediction (Breiman, 2001). RF can handle sparse datasets, data with missing values, as well as data that contain noise and errors (Ashraf et al., 2018).

LR is widely used for text classification problems (Lemon et al., 2003). LR can explain the relationship between dependent and independent variables. LR is a voting classifier that estimates the probability of an individual class and feeds it to the voting classifier which predicts the class with the highest probability (Rustam et al., 2019).

SVM is a powerful machine learning model used for non-linear classification and regression tasks (Bennett and Campbell, 2000). It makes use of boundary lines between the samples of each class, called 'hyperplanes'. Several hyperplanes may exist for a given data with respect to each class of data. The optimization aims at finding the best hyperplanes that maximize the distance between the class boundaries.

NB is based on the Bayesian theorem that calculates the probability of a given sample based on the prior probabilities. Despite being simple, NB often provides better results as compared to sophisticated classifiers. Assuming that the attributes are conditionally independent, a sample is attributed to a class with the highest posterior probability (Twala, 2010).

The performance evaluation of the proposed IRFTC is carried from twofold perspectives; binary classification and multi-class classification. For this purpose, a total of four datasets are used with two datasets for each task. The performance of the models is evaluated using accuracy, precision, recall, and F measure.

4.1.1. Binary class datasets

The SMS dataset used in this study is collected by Almeida et al. (2011) and Almeida (2016). It contains 'ham', and 'spam' classes for the records. A total of 5,574 samples are found in this dataset where 4,849 samples belong to the 'ham' class while 725 samples belong to the 'spam' class. The second dataset used for the binary classification task is the 'hate speech dataset' collected from Samoshyn (2020). It contains a total of 6,230 samples for 'hate speech' and 'neutral text' classes.

4.1.2. Multiclass datasets

The airline comments dataset is the first dataset used for experiments and it contains 4,390 tweets. It has three classes; 'negative', 'neutral', and 'positive' which shows the sentiments of the text posted on Twitter about airlines. The dataset is publicly available at Eight (2015). The hate speech dataset is the second dataset used for the multiclass classification task in this study and taken from Agarwal (2018). It contains a total of 5,960 text samples for 'offensive', 'neutral', and 'positive' classes. The number of samples and other details about these datasets are given in Table 7.

Table 7
Description of datasets.

Dataset	Class Type	Record	Classes	TFIDF vector dimensions
SMS dataset	Binary	5574	2(Spam, ham)	156
Hate and Offensive Language	Binary	6230	2(0,1)	140
US Airline Tweets	Multi Class	4390	3(Negative, Neutral, Positive)	162
Hate Speech	Multi Class	5960	3(-1,0,1)	152

Table 8
Results using SMS binary dataset.

Model	Accuracy	Precision	Recall	F measure
IRFTC	0.9221	0.8596	1.0000	0.9245
RF	0.9013	0.8333	1.0000	0.9091
LR	0.8913	0.8333	1.0000	0.9091
SVM	0.9040	0.8864	0.9750	0.9186
NB	0.8953	0.8333	1.0000	0.9091
DT	0.8843	0.8333	1.0000	0.9091

4.2. Results using binary class datasets

Table 8 shows the results of all machine learning models on the SMS binary class dataset. Clearly, the accuracy of the proposed IRFTC is better than both the traditional RF, as well as other machine learning models used for experiments. On average, IRFTC shows 2.08% accuracy improvement as compared to traditional RF. Similarly, F measure and recall results are also better than other models. IRFTC selects the optimal number of trees based on the iterative process of important feature selection which boosts its performance than the traditional RF.

Experimental results from the hate speech binary dataset are provided in Table Table 9. While the performance of all the models is sufficiently reasonable, the IRFTC shows remarkable improvement with a classification accuracy of 0.9396 which is significantly better than other models. Results on the hate speech dataset show better performance from IRFTC, although it is also the binary classification problem. However, the classification performance may change from one dataset to another depending on the number of samples used for training and the data distribution. The noteworthy point is the superior performance of the proposed IRFTC which is 6.28% better than traditional RF and 4.98% better than the best performing NB for hate speech dataset.

4.3. Performance evaluation using multiclass datasets

Besides the binary datasets, two multiclass datasets have also been used for experiments in the current study. Table 10 shows the results of machine learning models on US airline tweets which have three classes. Results suggest that IRFTC performs equally better with multiclass classification tasks where it obtains the

Table 9
Results of hate and offensive speech binary class dataset.

Model	Accuracy	Precision	Recall	F measure
IRFTC	0.9396	0.8772	1.0000	0.9346
RF	0.8805	0.8298	0.9750	0.8966
LR	0.8913	0.8333	1.0000	0.9091
SVM	0.8723	0.8333	1.0000	0.9091
NB	0.8953	0.8333	1.0000	0.9091
DT	0.8635	0.8298	0.9750	0.8966

Table 10
Results of Tweets on US airline multiclass dataset.

Model	Accuracy	Precision	Recall	F measure
IRFTC	0.9572	0.8571	0.9796	0.9143
RF	0.9430	0.8696	1.0000	0.9302
LR	0.8913	0.8333	1.0000	0.9091
SVM	0.8723	0.8333	1.0000	0.9091
NB	0.9162	0.8511	1.0000	0.9195
DT	0.9260	0.8696	1.0000	0.9302

Table 11
Results of hate speech multiclass dataset.

Model	Accuracy	Precision	Recall	F measure
IRFTC	0.9246	0.8236	0.9783	0.9024
RF	0.9013	0.8333	1.0000	0.9091
LR	0.8913	0.8333	1.0000	0.9091
SVM	0.9140	0.8864	0.9750	0.9286
NB	0.8953	0.8333	1.0000	0.9091
DT	0.8843	0.8333	1.0000	0.9091

highest accuracy of 0.9572 which is higher than all other machine learning models. The ensemble architecture of IRFTC helps to learn better than a single weak classifier. Also, it shows robustness to tackle the intra-class variation present in the text datasets. Consequently, its performance is second to none for the US airlines tweet dataset.

Experimental results for the hate speech multiclass dataset are provided in Table 11 which corroborates the superior performance of the proposed IRFTC model. Although the performance of IRFTC has not have a substantial difference from that of SVM whose accuracy is 0.9140, the IRFTC has the highest accuracy of 0.9246 which is still 1.15% better than the SVM.

4.4. Results using K-fold cross validation

To corroborate the efficacy of the proposed IRFTC model, this study performs 10-fold cross-validation results regarding the accuracy, precision, recall and F measure for each model with respect to each dataset used for experimentation. Results for four datasets used in this study are given in Tables 12–15 where Dataset 1 is the 'SMS binary dataset', Dataset 2 is the 'offensive speech dataset' while Dataset 3 and Dataset 4 represents the 'US airline dataset' and 'hate speech dataset', respectively. Results indicate that the proposed IRFTC performs superior to other machine learning models used in this study with increased accuracy and low standard deviation (SD). Also, the SD for other performance evaluation metrics like precision, recall and F measure is better than other models.

5. Conclusions and future work

Keeping in view the wide use of RF for text classification tasks due to its ability to work with imbalanced and noisy data and sparse feature distribution, this study enhances the performance of traditional RF. In this regard, this study proposes a lightweight improved Rf for text classification, IRFTC model to improve text

Table 12
K-fold cross validation results for SMS Binary Dataset (Dataset 1).

Model	Accuracy	SD	Precision	SD	Recall	SD	F Measure	SD
IRFTC	0.9256	0.0350	0.9078	0.0660	0.9092	0.0677	0.9085	0.0669
RF	0.9029	0.0525	0.8957	0.0905	0.9095	0.0467	0.9025	0.0616
LR	0.8941	0.0647	0.8887	0.0705	0.8857	0.0752	0.8872	0.0727
SVM	0.8954	0.0487	0.8931	0.0514	0.8921	0.0528	0.8926	0.0521
NB	0.8978	0.0492	0.8983	0.0488	0.8963	0.0505	0.8973	0.0496
DT	0.9089	0.0579	0.9071	0.0569	0.9041	0.0609	0.9056	0.0588

Table 13
K-fold cross validation results for Hate and Offensive Speech (Dataset 2).

Model	Accuracy	SD	Precision	SD	Recall	SD	F Measure	SD
IRFTC	0.9256	0.0350	0.9092	0.0549	0.8974	0.0950	0.9032	0.0696
RF	0.8576	0.0525	0.8953	0.0410	0.8884	0.0786	0.8919	0.0539
LR	0.8882	0.0647	0.8897	0.0595	0.8897	0.0595	0.8897	0.0595
SVM	0.8629	0.0487	0.8802	0.0365	0.8721	0.0545	0.8762	0.0437
NB	0.8887	0.0492	0.8872	0.0485	0.8862	0.0495	0.8867	0.0490
DT	0.8808	0.0579	0.8898	0.0639	0.8908	0.0641	0.8902	0.0640

Table 14
K-fold cross validation results for US airline dataset (Dataset 3).

Model	Accuracy	SD	Precision	SD	Recall	SD	F Measure	SD
IRFTC	0.9290	0.0185	0.9157	0.0351	0.9134	0.0396	0.9145	0.0372
RF	0.9203	0.0280	0.9126	0.0327	0.9144	0.0275	0.9135	0.0299
LR	0.8979	0.0456	0.8979	0.0456	0.8981	0.0454	0.8980	0.0455
SVM	0.8706	0.0483	0.8706	0.0483	0.8748	0.0458	0.8727	0.0470
NB	0.9005	0.0293	0.9005	0.0293	0.9007	0.0292	0.9006	0.0292
DT	0.9021	0.0409	0.9021	0.0409	0.9024	0.0406	0.9023	0.0408

Table 15
K-fold cross validation results for Hate Speech dataset (Dataset 4).

Model	Accuracy	SD	Precision	SD	Recall	SD	F Measure	SD
IRFTC	0.9202	0.0188	0.9185	0.0167	0.9165	0.0163	0.9175	0.0165
RF	0.9191	0.0344	0.9146	0.0305	0.9117	0.0314	0.9131	0.0309
LR	0.8892	0.0346	0.8875	0.0362	0.8865	0.0375	0.8869	0.0368
SVM	0.9019	0.0307	0.9024	0.0308	0.9029	0.0309	0.9027	0.0308
NB	0.8913	0.0410	0.8923	0.0400	0.8963	0.0383	0.8943	0.0391
DT	0.8937	0.0390	0.8929	0.0405	0.8930	0.0389	0.8930	0.0397

classification performance. The performance improvement is obtained using the feature importance and an optimal number of trees. Unimportant features are removed recursively with respect to their role in the classification accuracy. Similarly, the optimal number of trees is selected based on the strength and correlation-based good split. For performance evaluation, extensive experiments are performed using two binary and two multiclass datasets. Results suggest that IRFTC improves the performance of traditional RF by 6.28% for binary classification while 4.98% with respect to the best performer NB. On the other hand, IRFTC shows 1.50% better performance than traditional RF and 3.37% than the DT which is the second best performer after RF for the multiclass classification task. Results show that the proposed IRFTC shows equally superior performance with both binary and multiclass text classification. Furthermore, the performance on datasets from multiple domains proves its robustness and efficacy.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2019R1A2C1006159) and (NRF-2021R1A6A1A03039493).

References

R. Agarwal, Twitter hate speech, url: https://www.kaggle.com/vkrahul/twitter-hate-speech, 2018.
Almeida, T.A., Hidalgo, J.M.G., Yamakami, A., 2011. Contributions to the study of sms spam filtering: new collection and results. In: Proceedings of the 11th ACM symposium on Document engineering, pp. 259–262.
T.A. Almeida, Sms spam collection dataset, url: https://www.kaggle.com/uciml/sms-spam-collection-dataset, 2016.
Ashraf, I., Hur, S., Park, Y., 2018. Magio: Magnetic field strength based indoor-outdoor detection with a commercial smartphone. Micromachines 9 (10), 534.
Bennett, K.P., Campbell, C., 2000. Support vector machines: hype or hallelujah?. ACM SIGKDD explorations newsletter 2 (2), 1–13.
Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.
Chaudhary, A., Kolhe, S., Kamal, R., 2016. An improved random forest classifier for multi-class classification. Inform. Process. Agricul. 3 (4), 215–222.
Criminisi, A., Shotton, J., 2013. Decision forests for computer vision and medical image analysis. Springer Science & Business Media.
A. Cuzzocrea, S.L. Francis, M.M. Gaber, An information-theoretic approach for setting the optimal number of decision trees in random forests, in: 2013 IEEE International Conference on Systems, Man, and Cybernetics, IEEE, 2013, pp. 1013–1019.
T.G. Dietterich, Ensemble methods in machine learning, in: International workshop on multiple classifier systems, Springer, 2000, pp. 1–15.
F. Eight, Twitter us airline sentiment, url: https://www.kaggle.com/crowdflower/twitter-airline-sentiment, 2015.
Feng, W., Ma, C., Zhao, G., Zhang, R., 2020. Fsrf: An improved random forest for classification. In: 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA), IEEE, pp. 173–178.
Hosmer Jr, D.W., Lemeshow, S., Sturdivant, R.X., 2013. Applied logistic regression, Vol. 398. John Wiley & Sons.
Jacob, S.G. et al., 2015. Improved random forest algorithm for software defect prediction through data mining techniques. Int. J. Comput. Appl. 117 (23).
Kalaiselvi, B., Thangamani, M., 2020. An efficient pearson correlation based improved random forest classification for protein structure prediction techniques. Measurement 162, 107885.
Khalid, M., Ashraf, I., Mehmood, A., Ullah, S., Ahmad, M., Choi, G.S., 2020. Gbsvm: Sentiment classification from unstructured reviews using ensemble classifier. Appl. Sci. 10 (8), 2788.
T.M. Khoshgoftaar, M. Golawala, J. Van Hulse, An empirical study of learning from imbalanced data using random forest, in: 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007), Vol. 2, IEEE, 2007, pp. 310–317.
Kou, G., Yang, P., Peng, Y., Xiao, F., Chen, Y., Alsaadi, F.E., 2020. Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods. Appl. Soft Comput. 86, 105836.
Lakshmanaprabu, S., Shankar, K., Ilayaraja, M., Nasir, A.W., Vijayakumar, V., Chilamkurti, N., 2019. Random forest for big data classification in the internet of things using optimal features. Int. J. Mach. Learn. Cybern. 10 (10), 2609–2618.
P. Latinne, O. Debeir, C. Decaestecker, Limiting the number of trees in random forests, in: International workshop on multiple classifier systems, Springer, 2001, pp. 178–187.
Lemon, S.C., Roy, J., Clark, M.A., Friedmann, P.D., Rakowski, W., 2003. Classification and regression tree analysis in public health: methodological review and comparison with logistic regression. Ann. Behav. Med. 26 (3), 172–181.
Liu, X.-Y., Wu, J., Zhou, Z.-H., 2008. Exploratory undersampling for class-imbalance learning. IEEE Trans. Syst., Man, Cybern. Part B (Cybernetics) 39 (2), 539–550.
Loh, W.-Y., 2011. Classification and regression trees. Wiley interdisciplinary reviews: data mining and knowledge discovery 1 (1), 14–23.
Luo, C., Wang, Z., Wang, S., Zhang, J., Yu, J., 2015. Locating facial landmarks using probabilistic random forest. IEEE Signal Process. Lett. 22 (12), 2324–2328.
A. Mehmood, B.-W. On, I. Lee, I. Ashraf, G.S. Choi, Spam comments prediction using stacking with ensemble learning, in: Journal of Physics: Conference Series, Vol. 933, IOP Publishing, 2017, p. 012012.
Murphy, K.P. et al., 2006. Naive bayes classifiers. University of British Columbia 18 (60), 1–8.
Noble, W.S., 2006. What is a support vector machine?. Nature Biotechnol. 24 (12), 1565–1567.
Nugroho, K., Noersasongko, E., Fanani, A.Z., Basuki, R.S., et al., 2019. Improving random forest method to detect hatespeech and offensive word. In: 2019 International Conference on Information and Communications Technology (ICOIACT), IEEE, pp. 514–518.
T.M. Oshiro, P.S. Perez, J.A. Baranauskas, How many trees in a random forest?, in: International workshop on machine learning and data mining in pattern recognition, Springer, 2012, pp. 154–168.
Paing, M.P., Choomchuay, S., 2018. Improved random forest (rf) classifier for imbalanced classification of lung nodules. In: 2018 International Conference on Engineering, Applied Sciences, and Technology (ICEAST), IEEE, pp. 1–4.
Parida, U., Nayak, M., Nayak, A.K., 2021. News text categorization using random forest and naïve bayes. In: 2021 1st Odisha International Conference on Electrical Power Engineering, Communication and Computing Technology (ODICON), IEEE, pp. 1–4.
Paul, A., Mukherjee, D.P., 2015. Mitosis detection for invasive breast cancer grading in histopathological images. IEEE Trans. Image Process. 24 (11), 4041–4054.
Paul, A., Mukherjee, D.P., Das, P., Gangopadhyay, A., Chintha, A.R., Kundu, S., 2018. Improved random forest for classification. IEEE Trans. Image Process. 27 (8), 4012–4024.
Quadrianto, N., Ghahramani, Z., 2014. A very simple safe-bayesian random forest. IEEE Trans. Pattern Anal. Mach. Intell. 37 (6), 1297–1303.
Rustam, F., Ashraf, I., Mehmood, A., Ullah, S., Choi, G.S., 2019. Tweets classification on the base of sentiments for us airline companies. Entropy 21 (11), 1078.
A. Samoshyn, Hate speech and offensive language dataset, url: https://www.kaggle.com/mrmorj/hate-speech-and-offensive-language-dataset, 2020.
Twala, B., 2010. Multiple classifier application to credit risk assessment. Expert Syst. Appl. 37 (4), 3326–3336.
Umer, M., Ashraf, I., Mehmood, A., Ullah, S., Choi, G.S., 2021. Predicting numeric ratings for google apps using text features and ensemble learning. ETRI J. 43 (1), 95–108.
Xu, B., Guo, X., Ye, Y., Cheng, J., 2012. An improved random forest classifier for text categorization. J. Comput. 7 (12), 2913–2920.