Research Article

Multi-category news classification using Support Vector Machine based classifiers

Pooja Saigal[1] · Vaibhav Khanna[2]

Received: 19 December 2019 / Accepted: 14 February 2020 / Published online: 20 February 2020

Abstract
Support Vector Machine (SVM) and its variants are gaining momentum among the Machine Learning community. In this paper, we present a quantitative analysis between the established SVM based classifiers on multi-category text classification problem. Here, we are particularly interested in studying the behaviour of Least-squares Support Vector Machines, Twin Support Vector Machines and Least-squares Twin Support Vector Machines (LS-TWSVM) classifiers on News data. Since, all these are binary classifiers, they are extended using One-Against-All approach to handle multi-category data. The dataset is first converted into required format by performing preprocessing activities which involve tokenization and removing irrelevant data. The feature set is constructed as Term Frequency-Inverse Document Frequency matrix, so that representative vectors could be obtained for each document. Experimentally, we have compared the performance of each classification algorithm by performing simulations on benchmark UCI News datasets: Reuters and 20 Newsgroups. This paper shows that LS-TWSVM proves to be the best of all three, both in terms of accuracy and time complexity (training and testing).

Keywords SVM · Text categorization · LS-SVM · TWSVM · LS-TWSVM · TF-IDF · Tokenization · Stemming

1 Introduction

These days Internet is flooded with huge amount of text based entities that it becomes very difficult to manually categorize them. Tasks like categorizing email as spam or safe, patient's medical reports, academic papers based on their technical specification, insurance policies etc. have become cumbersome due to their huge volume. Thus, there is a need for some automatic technique which makes it easy and machine oriented, to classify text documents. This process of text classification can be achieved by the use of Machine Learning (ML) algorithms.

Text classification, also known as *Text Categorization*, is the method of categorizing and/ or sorting the text based entities into some predefined set of semantic categories or labels. Each entity can be designated as belonging to multiple, exactly one or no category (class or topic) at all. There are various techniques that are being used in the process of automatic text classification. The very first technique developed for this purpose was the use of Natural Language Processing (NLP) objectives with simple probabilistic models like Naive Bayes [1]. Here, the frequency or occurrence of each word is used as a feature for training the classifier. These NLP techniques were further replaced by more advanced and automatic method through the incorporation of Machine Learning Objectives in the task of automatic text classification. ML algorithms are divided into supervised, semi-supervised and unsupervised learning approaches. The supervised learning approach employees the method of training the system based on

✉ Pooja Saigal, pooja.saigal@vips.edu; saigal.pooja.in@gmail.com; Vaibhav Khanna, khannavaibhav21@gmail.com | [1]School of Information Technology, Vivekananda Institute of Professional Studies, New Delhi, India. [2]Applied AI and Data Science, AXA XL, Gurugram, Haryana, India.

output labels, and then tests the model on raw data. Whereas, unsupervised learning methods are used when it is difficult to get labeled entities and hence the models are trained without any labels. Semi-supervised Learning approach mediates between supervised and unsupervised approaches, as it involves large amount of unlabeled data and a small portion of labeled data for training purpose.

The Support Vector Machine (SVM) [2–4] is a popular supervised learning algorithm used for pattern recognition and regression analysis, which later found its application in the field of text classification. SVM has decent differentiating capabilities in various applications like particle identification, face recognition, image classification etc. The main idea behind SVM is to obtain an optimal hyperplane which distinguishes the entities belonging to two classes as positive or negative. In the recent times, there have been several improvements in the design of traditional SVM such as Lagrangian Support Vector Machine (LSVM) [5], Proximal Support Vector Machine (P-SVM) [6], Least Square Support Vector Machine (LS-SVM) [7]. These model generate two *parallel* hyperplanes and the data points are classified according to their distance from these hyperplanes. Mangasarian et al. proposed Generalized Eigenvalue Proximal Support Vector Machine (GEPSVM) [8], which solves a pair of generalized Eigen-value problems to generate two nonparallel, proximal hyperplanes for the two classes. Based on the idea of GEPSVM, Jayadeva et al. proposed a Twin Support Vector Machine (TWSVM) [9], which solves a pair of Quadratic Programming Problems (QPPs), for generating the two nonparallel hyperplanes. It is in contrast to the traditional SVM that solves a single complex QPP to generate a single maximum-margin separating hyperplane. TWSVM has gained lot of popularity in the last decade and many variations of TWSVM have been proposed [10–16]. A variation of TWSVM, Least Square Twin Support Vector Machine (LS-TWSVM) [17], finds its application in text classification problem. LS-TWSVM incorporates the positives of both LS-SVM and TWSVM.

Joachims et al. [18] proved the suitability of SVM for text classification but it is found that SVM based systems face difficulties during information retrieval as it does not consider the semantic relations between the data terms. Therefore, the approach for Latent Semantic Indexing(LSI) based Least Square Support Vector Machine was incorporated. LSI is a document indexing technique in which the model is generated based on word occurrences in that document. LSI algorithm formulates a matrix representation of the corpus, in which rows corresponds to words in the vocabulary and columns corresponds to words in the document. Each entry in the matrix is thus a weighted frequency of corresponding term in corresponding document, eliminating the impact of frequently occurring terms. LSI coefficient based LS-SVM proved to an efficient method for Text classification. The use of LS-SVM for Text Categorization was accomplished by Mitra et al. in 2004 [19] where they presented an analysis of Least Square Support Vector machine with LSI for feature extraction. The aim was to compare the results obtained from LS-SVM with K-Nearest neighbor (KNN) and Naive Bayesian (NB) classifier and portray the potential and relevance of using LS-SVMs as the intelligent classifying agent for text classification. The robustness of this system enabled the classification of noisy text titles with a high degree of precision. He et al. [20] compared three machine learning methods, namely *k*-Nearest Neighbor (kNN), Support Vector Machines (SVM), and Adaptive Resonance Associative Map (ARAM) for Chinese document classification. Lee et al. [21] proposed enhanced SVM framework for text document classification.

In this paper, we compare the results for LS-SVM, TWSVM and LS-TWSVM for multi-category News classification problem. The comparison is done in terms of generalization ability and time complexity of the individual algorithms on benchmark UCI datasets. The organization of the paper is as follows: Sect. 2 discusses about the different SVM based classifiers used for text classification namely LS-SVM, TWSVM, LS-TWSVM. Section 3 presents techniques used for retrieving relevant information from the unstructured dataset i.e. it explains the steps used for preprocessing of dataset. Section 4 presents the proposed algorithm for News classification using SVM-based classifiers. Section 5 talks about News datasets used for performing experiments and the numerical findings observed during the simulations. Finally, the paper is concluded in Sect. 6.

2 SVM-based classifiers

SVM has attracted a lot of researchers in the last two decades. SVM is initially proposed for classification and later it is extended to handle regressions problems also. Since, SVM solves a convex quadratic problem, it converges to a global solution in a definite time. SVM is a generic classifier and can be applied to solve numerous problems in different domains. Recently, SVM-based classifiers have been used for text classification. In the following section, we present the mathematical formulation of three SVM-based classifiers, which are later used for text classification.

2.1 Twin Support Vector Machine

Twin Support Vector Machines (TWSVM) forms a part of the new emerging machine learning approaches, which are being applied in the field of text classification problems. Twin Support Vector Machine (TWSVM) introduced

by Jayadeva et al. [9] is based on the basic concepts of traditional SVM and is developed on the lines of Generalized Eigenvalue Proximal Support Vector Machine (GEPSVM) [6]. TWSVM generates two nonparallel hyperplanes for two class problems. The two nonparallel hyperplanes are given as:

$$x^T w_1 + b_1 = 0 \text{ and } x^T w_2 + b_2 = 0. \tag{1}$$

Here, w_i, $i = \{1, 2\}$ is the normal vector to the ith hyperplane and b_i is the corresponding bias. These proximal hyperplanes are obtained by solving two smaller SVM-type problems. TWSVM solves a pair of QPPs instead of a single complex QPP as in the case of conventional SVM. Figure 1 shows the kind of hyperplanes generated by SVM and non-parallel hyperplane classifiers.

TWSVM, being a binary classifier, classifies the data sample entities into two classes namely + 1 and − 1. Let A and B be the two matrices of dimensions $m_1 \times n$ and $m_2 \times n$ respectively, such that they contain m_1 and m_2 data points in n-dimensional real space region $\mathbb{R}^n$ from two different classes. The TWSVM classifier is obtained by solving the following pair of QPP's: (TWSVM1):

$$\min_{w_1, b_1, \xi_2} \frac{1}{2}(Aw_1 + e_1 b_1)^T (Aw_1 + e_1 b_1) + c_1 e_2^T \xi_2$$
$$\text{subject to } -(Bw_1 + e_2 b_1) + \xi_2 \geq e_2, \quad \xi_2 \geq 0 \tag{2}$$

(TWSVM2):

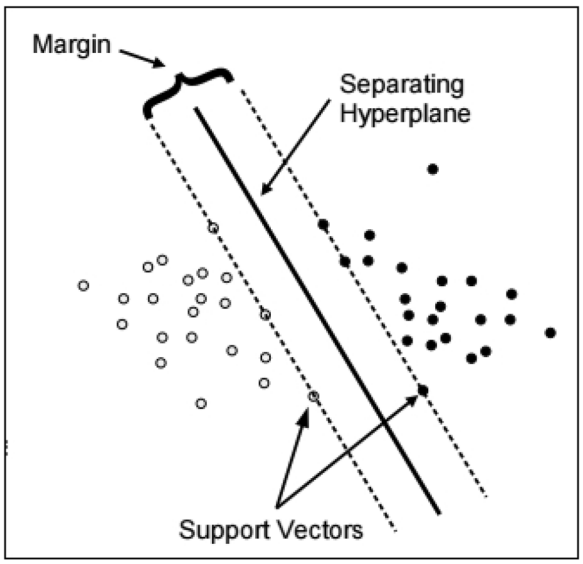
$$\min_{w_2, b_2, \xi_1} \frac{1}{2}(Bw_2 + e_2 b_2)^T (Bw_2 + e_2 b_2) + c_2 e_1^T \xi_1$$
$$\text{subject to } (Aw_2 + e_1 b_2) + \xi_1 \geq e_1, \quad \xi_1 \geq 0. \tag{3}$$

The constant $c_1 > 0$ ($c_2 > 0$) is the trade-off factor between error vector ξ_2 (ξ_1) due to the other class and distance of hyperplane from its own class; and e_1 and e_2 are vectors of ones of appropriate dimensions. The first term in the objective function of (2) or (3) is the sum of squared distances of the hyperplane to the data points of its class. Thus, minimizing this term tends to keep the hyperplane proximal to the points of one class (say class + 1) and the constraints require the hyperplane to be at unit distance from the points of other class (say class − 1); error variables ξ_1 and ξ_2 are used to measure the violation wherever the hyperplane is less than unit distance away from data points of other class. The second term of the objective function minimizes the sum of error variables, thus it attempts to minimize misclassification due to points belonging to other class.

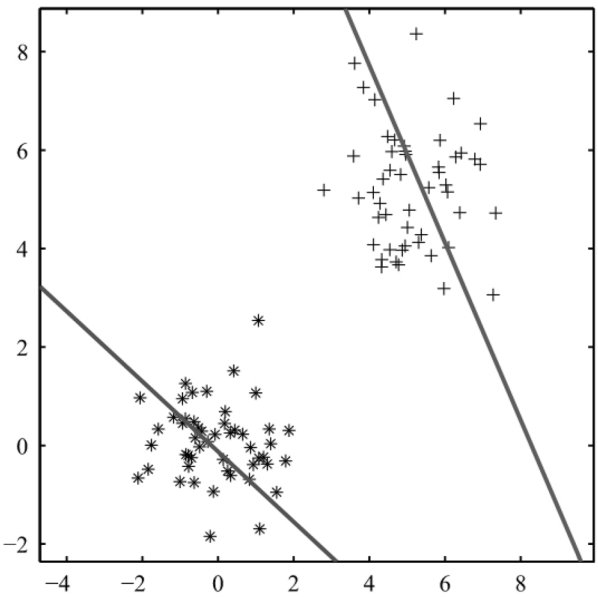
The solutions of the above mentioned pair of QPPs is determined by solving their dual problems. The dual optimization problem is obtained by solving Karush–Kuhn–Tucker (KKT) necessary and sufficient optimality conditions [22]. The Wolfe dual [11] of (TWSVM1) is given as follows: (DTWSVM1):

$$\max_{\alpha} e_2^T \alpha - \frac{1}{2}\alpha^T G(H^T H)^{-1} G^T \alpha$$
$$\text{subject to } 0 \leq \alpha \leq c_1. \tag{4}$$

Similarly, we consider (TWSVM2) and obtain its dual as (DTWSVM2):



(a) Maximum margin separating hyperplane generated by SVM or LS-SVM between two classes



(b) Proximal hyperplanes generated by TWSVM or LS-TWSVM for two classes

Fig. 1 Hyperplanes generated by SVM-based classifiers

$$\max_{\gamma} e_1^T\gamma - \frac{1}{2}\gamma^T P(Q^TQ)^{-1}P^T\gamma \quad (5)$$
subject to$0 \le \gamma \le c_2$.

Here, $H = [A \;\; e_1]$, $G = [B \;\; e_2]$, $P = [A \;\; e_1]$, $Q = [B \;\; e_2]$. The augmented vectors $u = [w_1, b_1]^T$ and $v = [w_2, b_2]^T$ are given by

$$u = -(H^TH)^{-1}G^T\alpha \quad (6)$$

and

$$v = (Q^TQ)^{-1}P^T\gamma. \quad (7)$$

Here, $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_{m_2})^T$ and $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{m_1})^T$ are Lagrange multipliers.

In a nutshell, TWSVM comprises of a pair of QPPs such that, in each QPP, the objective function corresponds to a particular class and the constraints are determined by patterns of the other class. Thus, TWSVM gives rise to two smaller sized QPPs. In (TWSVM1), patterns of class +1 are proximal to the hyperplane $x^Tw_1 + b_1 = 0$. Similarly, in (TWSVM2), patterns of class −1 lie around the hyperplane $x^Tw_2 + b_2 = 0$. Also, TWSVM is approximately four times faster than SVM. This is because the complexity of the usual SVM is no more than m^3, where m is the total number of samples and TWSVM solves two problems, namely, (1) and (2), each of which is roughly of size $(m/2)$. Thus, the ratio of run-time is approximately $[(m^3)/(2 \times (m/2)^3)] = 4$.

A new data sample $x \in \mathbb{R}^n$ is assigned to class $r (r = 1, 2)$, depending on which of the two planes given by (1) it lies closer to, i.e.

$$x^Tw_r + b_r = \min_{l=1,2} \frac{|x^Tw_l + b_l|}{\|w_l\|}, \quad (8)$$

where $|.|$ is the absolute distance of point x from the plane $x^Tw_l + b_l = 0,\ l = 1, 2$.

In order to extend the results to non-linear classifiers, the kernel-generated surfaces are considered instead of planes, as given in (9) and (10).

$$K(x^T, C^T)u_1 + b_1 = 0, \quad (9)$$

and

$$K(x^T, C^T)u_2 + b_2 = 0 \quad (10)$$

where $C^T = [A \quad B]^T$ and K is an appropriately chosen kernel. The primal QPPs of the nonlinear TWSVM corresponding to the surfaces (9) and (10) are given by (11) and (12) respectively.

(KTWSVM1):

$$\min_{u_1, b_1, \xi_2} \frac{1}{2}\|(K(A, C^T)u_1 + e_1b_1)\|^2 + c_1e_2^T\xi_2 \quad (11)$$
subject to $-(K(B, C^T)u_1 + e_2b_1) + \xi_2 \ge e_2, \xi_2 \ge 0,$

(KTWSVM2):

$$\min_{u_2, b_2, \xi_1} \frac{1}{2}\|(K(A, C^T)u_2 + e_2b_2)\|^2 + c_2e_1^T\xi_1 \quad (12)$$
subject to $(K(B, C^T)u_2 + e_1b_2) + \xi_1 \ge e_1, \xi_1 \ge 0,$

where parameters $c_1 > 0$ and $c_2 > 0$. The Wolfe duals of KTWSVM1 and KTWSVM2 are given by (13) and (14) respectively.

(KDTWSVM1):

$$\max_{\alpha} \; e_2^T\alpha - \frac{1}{2}\alpha^T R(S^TS)^{-1}R^T\alpha \quad (13)$$
subject to $0 \le \alpha \le c_1$,

where $S = [K(A, C^T) \quad e_1], R = [K(B, C^T) \quad e_2]$.

(KDTWSVM2):

$$\max_{\gamma} \; e_1^T\gamma - \frac{1}{2}\gamma^T L(N^TN)^{-1}L^T\gamma \quad (14)$$
subject to $0 \le \gamma \le c_2$,

where $L = [K(A, C^T) \quad e_1], N = [K(B, C^T) \quad e_2]$.

Once (KDTWSVM1) and (KDTWSVM2) are solved to obtain the surfaces (9) and (10), a new pattern $x \in \mathbb{R}^n$ is assigned to class 1 or class − 1in a manner similar to the linear case.

2.2 Least Square Support Vector Machines (LS-SVM)

Sukyens et al. [7] proposed the least squares version of SVM and termed it as Least Squares Support Vector Machines (LS-SVM). On the lines of SVM, LS-SVM obtains the maximum-margin separating hyperplane, given as:

$$x^Tw + b \ge 1, \quad (15)$$

which is at least unit distance away from the patterns of both the classes. Here, $(w,\ b)$ are the parameters of the normal to the optimal hyperplane. The optimization problem of LS-SVM is quite different from that of traditional SVM. LS-SVM considers equality constraints instead of inequality constraints. In the objective function, L_2-norm of error is considered. As a consequence, its solution is obtained by solving a system of linear equations instead of solving quadratic programming problem (QPP). The optimization problem for LS-SVM is formulated as:

(LS-SVM:)

$$\min_{w,b,\xi} \frac{1}{2}w^T w + \frac{c_1}{2}\xi^2, \quad \text{subject to } Y[w^T X + b] = e - \xi, \tag{16}$$

where, X represents the data matrix that contains all the data points. The dimensions of X are $(m \times n)$, here m is the number of training points and n is the feature dimension in real space $\mathbb{R}^n$. The slack vector ξ captures the amount of violations of constraints ad c_1 is the penalty parameter. $Y \in \{+1, -1\}$ is the vector of class labels and e is a vector of ones of appropriate dimension. Instead of solving the dual problem, as done in SVM, the solution is obtained by solving the following system of linear equation:

$$\begin{bmatrix} I & 0 & 0 & -Z^T \\ 0 & 0 & 0 & -Y^T \\ 0 & 0 & c_1 I & -I \\ Z & Y & I & 0 \end{bmatrix} \begin{bmatrix} w \\ b \\ \xi \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ e \end{bmatrix} \tag{17}$$

where, $Z=[X^T Y_1; \ldots; X^T Y_N]$, $Y = [Y_1; \ldots Y_N]$, $e = [1; \ldots 1]$, $\xi = [\xi_1; \ldots; \xi_N]$, $\alpha = [\alpha_1; \ldots; \alpha_N]$, and the equation (17) can be further solved as:

$$\begin{bmatrix} 0 & -Y^T \\ Y & ZZ^T + c_1^{-1} I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ e. \end{bmatrix} \tag{18}$$

LS-SVM is proved to have better generalization ability and lower computational cost than the traditional SVM. To classify linearly inseparable datasets, we can make use of kernel trick, as discussed for TWSVM.

2.3 Least Squares Twin Support Vector Machine

Following the success of TWSVM, Kumar et al. [23] proposed least squares version of TWSVM for binary classification and called it as Least Squares Twin Support Vector Machine (LS-TWSVM). This formulation is developed on the lines of LS-SVM and leads to extremely simple and fast algorithm for generating binary classifiers based on two nonparallel hyperplanes. LS-TWSVM solves two modified primal problems of TWSVM. The hyperplanes are obtained by solving two systems of linear equations as opposed to solving two QPPs along with two systems of linear equations in TWSVM. The optimization problems for LS-TWSVM are given as:

(LS-TWSVM1):

$$\min_{w_1,b_1,\xi_2} \frac{1}{2}(Aw_1 + e_1 b_1)^T (Aw_1 + e_1 b_1) + \frac{c_1}{2}\xi_2^T \xi_2 \quad \text{subject to } -(Bw_1 + e_2 b_1) + \xi_2 = e_2 \tag{19}$$

(LS-TWSVM2):

$$\min_{w_2,b_2,\xi_1} \frac{1}{2}(Bw_2 + e_2 b_2)^T (Bw_2 + e_2 b_2) + \frac{c_2}{2}\xi_1^T \xi_1 \quad \text{subject to } (Aw_2 + e_1 b_2) + \xi_1 = e_1. \tag{20}$$

The above equations use L_2-norm of the slack variables ξ_i and the inequality constraints of TWSVM are replaced by equality constraints. This simplifies the solution of (19) as a system of linear equations. On substituting the value of ξ_1, the equation is transformed as:

$$\min_{w_1,b_1} \frac{1}{2}||(Aw_1 + e_1 b_1)||^2 + \frac{c_1}{2}||Bw_1 + e_2 b_1 + e_2||^2. \tag{21}$$

The above mentioned problems are solved in their primal form, in contrast to TWSVM. By setting the gradients, with respect to w_1 and b_1, equal to zero and by rearranging the equations in matrix form for the variables w_1 and b_1, following equations are obtained:

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = -\left(G^T G + \frac{1}{c_1} H^T H\right)^{-1} G^T e_2 \tag{22}$$

Similarly, the solution of QPP (20) can be explained as follows:

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = \left(H^T H + \frac{1}{c_2} G^T G\right)^{-1} H^T e_1. \tag{23}$$

Here, the augmented matrices G and H have their usual meaning as described before. After obtaining the pair of nonparallel, proximal hyperplanes, a new data point x, belonging to R^n real space, is assigned the label of class $+1$ or -1 depending on the criteria of minimum distance between the planes and the data point x. The minimum perpendicular distance from the two hyperplanes is represented by $|x^T w_1 + b_1|$ and $|x^T w_2 + b_2|$ respectively. LS-TWSVM can be extended to handle linearly inseparable data by using kernel trick, as discussed for TWSVM.

3 Preprocessing of dataset

The process of *Text Categorization* involves categorizing the text-based documents into predefined classes. But this task involves a level of complexity in the sense that the dataset contains a large number of documents which are required to be classified into predefined categories. For the purpose of completion of categorization process, two advanced techniques are implemented on the dataset, called extraction and preprocessing. Figure 2 shows the steps required for preprocessing of documents.

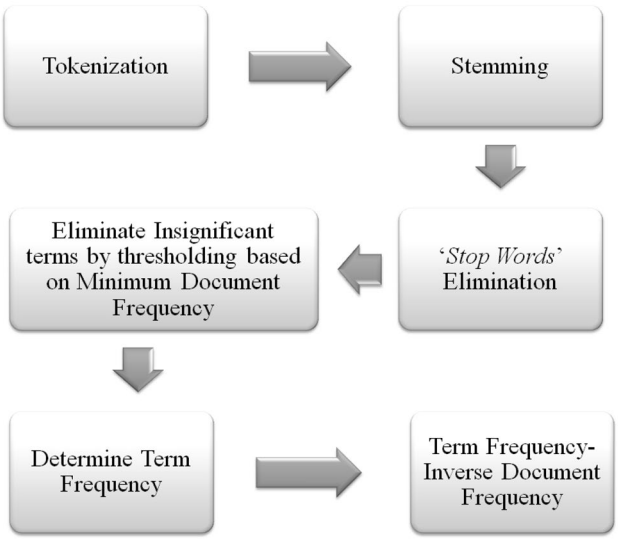


Fig. 2 Different steps used in preprocessing of the document

3.1 Data extraction

The dataset used in the process of text classification contains the sample data in different formats. For performing numerical experiments, there is a need to use the samples that suit our system's. So, we chose Reuters and 20 Newsgroups datasets for performing numerical experiments. In these datasets, the documents are enclosed in tags and the structure of the document includes attributes like Topics, Places, Companies, Title, Organizations and Body.

3.2 Preprocessing

The sample dataset extracted for the purpose of text classification process include a large number of sample documents and each document in itself contains a huge volume of words, extending up to 1000 words per document. These words can further be categorized into two classes namely relevant words and irrelevant words. The presence of irrelevant words makes the process of classification complex. Thus, a preprocessing of each document in the dataset is required. The preprocessing of documents involves following sequence of steps.

3.2.1 Tokenization, stemming and elimination of stop words

Tokenization is the first preprocessing step of text classification. It can be explained as a process of transforming a text based entity into a list containing tokens. Token represents a single individual entity in a document. Tokens are classified into various classes, and each word in document is allotted to a unique class of tokens. The next step of preprocessing is called Stemming. It is a crucial step, as it helps in reducing the complexity of the document to a great extend. The process of stemming is defined as storing the derived words in their base or root form only i.e. storing only a single base word for various different forms of that word present in the document. Stop words are the words that do not pertain to a specific purpose in classification process, but only results in increasing the volume of tokens in the documents. The words that occur too commonly in documents like "in", "the", "a", "is" are all insignificant, as they do not have any relevance in the classification process. There is a predefined list of about 2000 stop words, which create a class of words that are required to be removed from the documents.

3.2.2 Minimum document frequency

Document frequency measures the relevance of a particular document for a given term. It is the number of times a given term t appears in a document d within a larger search index. Minimum Document Frequency (MinDF) is used for removing less frequently occurring terms from the document. Generally, the default value for MinDF is 1 which means that remove all the terms that occur in only one document. Similarly, $MinDF = 5$ removes the terms that appear in only 5 or less documents in the dataset. Such terms are treated as insignificant and do not have much influence on the classification process.

3.2.3 Term frequency

Term frequency (TF) is commonly used in Natural Language Processing (NLP), Text Mining etc. that measures how frequently a term occurs in a document. Here, the terms refer to tokens. Counting the occurrences of a term is not sufficient as the count largely depends on the size of the document. A large document would always give more count for the terms. To normalize this count, it is divided by the total number of terms in the document. So, term frequency for a term t in a document D is given as

$$TF(t, D) = \frac{\text{Number of times term t appears in a document}}{\text{Total number of terms in the document}} \quad (24)$$

There are many ways to normalize TF like dividing by the maximum term frequency in a document. Thus, higher the

value of $TF(t, D)$ for a particular term, more is its relevance in the process of classification. Here, a threshold value is specified below which all the tokens are considered insignificant.

3.3 Feature set generation: Term Frequency-Inverse Document Frequency (TF-IDF)

Once the token glossary is created, weights are assigned to tokens based on their relevance in the classification process. Term Frequency-Inverse Document Frequency (TF-IDF) [24] is a way of assigning weights to each term in a document and is based on the term's frequency (TF) and inverse document frequency (IDF). The product of these two terms is called the weight of the term. IDF determines the inverse probability of occurrence of a term in a document. The terms with higher TF-IDF weight scores are considered to be more significant. It check hoe relevant the term is, with respect to the whole corpus (dataset). Let, D be a collection of Documents, t be a particular term and $d \in D$ a specific document, then weight of t in d is calculated as:

$$W_{t,d} = TF(t, d) * log(|D|/DF(t, D)), \quad (25)$$

Here, $TF(t, d)$ represents term frequency of t in document d and IDF is given by $log(|D|/DF(t, D))$ where $DF(t, D)$ is the document frequency i.e. number of documents that contain term t in a corpus D. Here, $|D|$ represents the total number of documents.

The role of TF-IDF [25] in the process of preprocessing can be explained by the concept that at some instance, the frequency of a particular term in a document is very high but its relevance in contrast is very low. For instance, in a dataset relating to India, the term 'India' will occur multiple times but its significance in classifying the documents is negligible. Thus, the inverse document frequency, denoted as $IDF(t, D)$ is the rate of determining that whether the term is rare or common in the dataset. Thus, a common term has less information for classifying the documents while a rare term may have much more information about the dataset.

4 Algorithm for news categorization using SVM-based classifiers

In this section, we present the process of News categorization using SVM-based classifiers. This algorithm represents a generic approach to classify multi-category text data

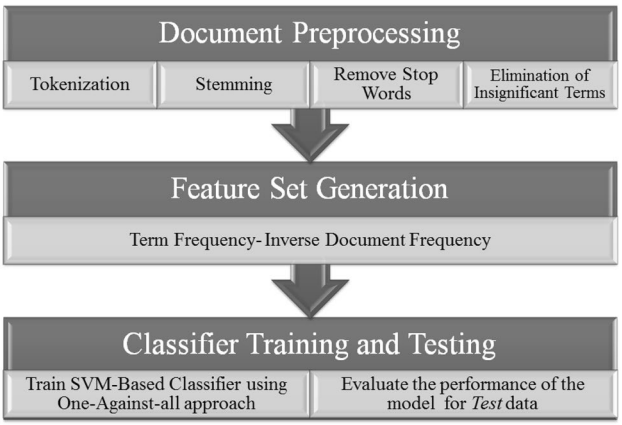


Fig. 3 Text categorization using SVM-based classifiers

using single hyperplane or pair of nonparallel hyperplanes classifiers. But for this work, our focus is on News categorization using LS-SVM, TWSVM and LS-TWSVM with One-Against-All [26] approach for multi-category extension of SVM. Figure 3 graphically shows the steps for classifying News dataset with the help of SVM-based classifiers.

SVMs have been quite popular as binary classifiers and there is a need to extend the same to multi-category classification problems. The two most popular approaches for multi-class SVMs are One-Against-All (OAA) and One-Against-One (OAO) support vector machines [26]. OAA-SVM implements a series of binary classifiers where each classifier separates one class from rest of the classes. But this approach leads to biased classification due to huge difference in the number of samples. For a K-class classification problem, OAA-SVM requires K binary SVM comparisons for each test data. In case of OAO-SVM, the binary SVM classifiers are determined using a pair of classes at a time. So, it formulates upto $(K * (K - 1))/2$ binary SVM classifiers, thus leading to increase in computational complexity. Also, directed acyclic graph SVMs (DAGSVMs) are proposed in [26], in which the training phase is the same as OAO-SVMs i.e. solving $(K * (K - 1))/2$ binary SVMs, however its testing phase is different. During testing phase, it uses a rooted binary directed acyclic graph which has $(K * (K - 1))/2$ internal nodes and K leaves.

In this work, we extended the three binary classifiers using OAA approach. LS-SVM is extended in similar manner as suggested for OAA-SVM [26]. The process is explained in Algorithm 1.

Input : Training data $X = \{X_1, X_2, ..., X_m\}$ with labels $Y \in \{1..K\}$ and Test data X_{test}. Here, K is the number of classes in the dataset.
Output : Class Labels for test data Y_{test}.
Process:
1. Select the regularization, kernel parameters for the classifier.
2. Construct K LS-SVM models.
   a. The i^{th} $i = \{1, .., K\}$ LS-SVM model is trained with all of the samples in the i^{th} class with positive labels, and all other samples with negative labels.
   b. Use (17) to generate the i^{th} hyperplane.
3. Test samples x_{text} is assigned the class label which has the largest value of the decision function

$$\text{class of } x_{test} = \arg \max_{i=1,..,K} ((w^i)^T x + b^i). \tag{28}$$

Algorithm 1: Multi-category News categorization using LS-SVM classifier

For nonparallel hyperplane classifiers i.e. OAA-TWSVM and OAA-LS-TWSVM, the OAA algorithm solves *K* QPPs, one for each class, so that we obtain $2 * K$ nonparallel hyperplanes for K classes. Here, we construct a TWSVM or LS-TWSVM classifier. For *i*th classifier, we solve one QPP taking *i*th class samples (represented by *A*) with positive labels and remaining samples as other class *B* with negative labels. By using this methodology, we determine the hyperplane for the *i*th class. The unbalance problem of exemplars existing in *i*th classifier is tackled by choosing the proper penalty parameter (c_1) for the *i*th class. Algorithm 2 explain the steps of text (News) classification using OAA extension of nonparallel hyperplane classifiers.

Input : Training data $X = \{X_1, X_2, ..., X_m\}$ with labels $Y \in \{1..K\}$ and Test data X_{test}. Here, K is the number of classes in the dataset.
Output : Class Labels for test data Y_{test}.
Process:
1. Select the regularization, kernel parameters for the classifier.
2. Train the classifier model using *Training data* X with OAA approach. Generate K classifier models. For each class $i = \{1..K\}$, create a positive class from its own samples and the other samples (belonging to rest of the $(K-1)$ classes) constitute the negative class. Obtain hyperplanes for these K positive classes using the datasets, thus created.
   a. For TWSVM as the classifier, use (6)-(7) or their kernel versions, to generate the nonparallel hyperplanes.
   b. For LS-TWSVM, use (22)-(24) to generate the proximal hyperplanes.
3. Use *minimum distance from hyperplanes* criteria, to classify the test data X_{test}.

Algorithm 2: Multi-category News categorization using SVM-based nonparallel hyperplane classifiers

5 Experiments

In order to prove the usability and efficacy of the established methods- LS-SVM [7], TWSVM [9] and LS-TWSVM [23] for News classification, we performed experiments on Reuters and 20 Newsgroups datasets. All the experiments are performed in MATLAB version 8.0 under Microsoft Windows environment on a machine with 3.40 GHz CPU and 8 GB RAM. The training-test datasets are created using 5-fold cross validation [27] and using One-Against-All multi-category approach. The classification accuracy is reported as the average accuracy over 5-folds, where 'Accuracy' is given by

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \tag{27}$$

Here TP, TN, FP, and FN are the number of true positive, true negative, false positive and false negative respectively.

To statistically evaluate the classification results, another evaluation criteria is used: *F*1 score, which is determined as

$$F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{28}$$

where

$$Precision = \frac{TP}{TP + FP}, \tag{29}$$

$$Recall = \frac{TP}{TP + FN}. \tag{30}$$

Here, *TP*, *FP*, *TN*, *FN* are true-positive, false-positive, true-negative and false-negative respectively.

5.1 Dataset description

The following section discusses about the benchmark UCI News datasets used for classification in order the compare different models.

5.1.1 Reuters-21578

The Reuters-21578[1] documents appeared in the Reuters Newswire in 1987. The documents were assembled and

[1] In http://www.daviddlewis.com/resources/testcollections/reuters21578.

indexed with categories by Reuters Ltd. and Carnegie groups. In 1990, the documents were made available by David D. Lewis of Information Retrieval Laboratory for various research purposes. The data set comprises of 21578 News articles, each belonging to one or more categories. In this dataset, all data files are named as reut2-*.sgm where * is varying from 000 to 021. The .sgm file format is like .xml file format. Every sample entry is in a tag named Reuters. For this comparative study, we only choose those documents that have unique class and every class should have at least one sample in Train set and one sample in Test set. Out of the 135 potential categories, only 90 categories have at least one training and one testing documents. The documents were then preprocessed to apply various machine learning algorithms. For these 21,578 documents, only the documents containing the four most popular topics were selected for training and testing purposes. The four topics are 'earn (earning)', 'acq (corporate acquisitions)', 'money-fx (money market)' and 'grain' categories.

5.1.2 20 Newsgroups

The 20 Newsgroups[2] dataset was collected by Lang. This corpus is a collection of approximately 20,000 articles taken from the Usenet Newsgroups that are distributed evenly across 20 different Newsgroups. This dataset contrasts from other datasets as it includes large vocabulary and words that have more meaning. The 20 Newsgroups dataset is much simpler than Reuters-21578. Every directory is a category and every file under the directory is a simple entry. At the beginning of every file, there are some lines of meta data of that file which will not be considered for training. So, only the body part after the lines of meta information will be extracted. Unlike other corpora, 20 Newsgroups data does not have standard training and testing sets. For every part, there is a list of text and labels. The document is then preprocessed and passed to machine learning classifiers for evaluation of result. Form these 20 Newsgroups we have used the four most important Newsgroups namely 'alt.atheism', 'talk.religion.misc', 'comp.graphics', 'sci.space' for the purpose of training and testing.

5.2 Experimental results on Reuters-21578

The three classifiers—LS-SVM, TWSVM and LS-TWSVM, are trained to classify the News documents of Reuters-21578, belonging to four classes. Here, the binary classifiers are extended using OAA approach to handle multi-category data. Table 1 shows the confusion matrix for the four classes when Reuters-21578 dataset is classified using LS-SVM. Here, it is observed that there is large number of diagonal entries as compared to the non-diagonal entries. This indicates that the classifier is able to correctly classify most of the News documents. It is also observed that the non-diagonal entries constitute 13.33% of the total testing documents so the classification results for 'Grain' are not satisfactory. These results could be improved with other classifiers.

Table 1 LS-SVM Reuters confusion matrix

Confusion matrix	Acq	Earn	Money-fx	Grain
Acq	726	50	0	0
Earn	47	1268	0	0
Money-fx	71	4	127	0
Grain	161	4	0	70

Table 2 TWSVM Reuters confusion matrix

Confusion matrix	Acq	Earn	Money-fx	Grain
Acq	754	24	1	1
Earn	26	1290	0	9
Money-fx	6	0	197	2
Grain	4	5	1	208

Table 3 LS-TWSVM Reuters confusion matrix

Confusion matrix	Acq	Earn	Money-fx	Grain
Acq	772	8	0	0
Earn	25	1300	0	0
Money-fx	4	2	199	0
Grain	2	2	0	214

Table 2 shows confusion matrix with TWSVM classifier. There is significant improvement in classification results, which is indicated by higher number of diagonal entries as compared to non-diagonal entries. Similar trend is observed in Table 3, which shows higher figures for diagonal entries. Both the classifiers, TWSVM as well as LS-TWSVM, are able to correctly classify most the test documents belonging to 'Grain' and other three classes. This indicates that the multi-category classification results obtained by nonparallel hyperplane classifiers are more accurate than LS-SVM which is experimentally proved by the classification accuracy results given in Fig. 4. It shows

[2] http://www.ics.uci.edu/.

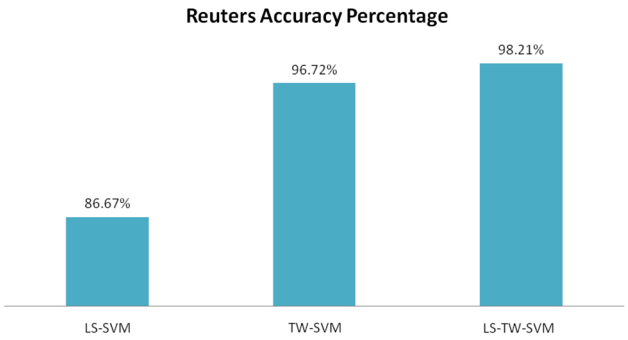


Fig. 4 The classification accuracy for Reuters with all three classifiers

that LS-TWSVM achieves the highest classification accuracy of 98.21% for Reuters dataset. Figure 4 shows the accuracy for Reuters dataset with all three classifiers and it is noted that LS-TWSVM outperforms the other two classifiers.

To statistically compare the performance of three classifiers, we determined precision, recall and *F*1 score for Reuters-21578 dataset, for each document category. The results are reported in Table 4. Here, the best results are shown in bold.

5.3 Experimental results on 20 Newsgroups

In this section, we present the classification results obtained for 20 Newsgroups dataset. Table 5 shows confusion matrix for this dataset, when classified with LS-SVM. The matrix although has more diagonal entries but the number of non-diagonal entries is also significant here. This indicates that many documents are misclassified and there is huge scope of improvement. LS-SVM is not able to give good results for 'talk.religion.misc' class, where more that half of the test documents are misclassified. To compare the performance of all three classifiers, the classification results are further obtained for this dataset using

Table 5 LS-SVM 20 Newsgroups confusion matrix

Confusion matrix	alt.atheism	talk.religion.misc	comp.graphics	sci.space
alt.atheism	238	12	2	0
talk.religion.misc	110	101	4	2
comp.graphics	38	0	352	2
sci.space	36	0	4	359

Table 6 TWSVM 20 Newsgroups confusion matrix

Confusion matrix	alt.atheism	talk.religion.misc	comp.graphics	sci.space
alt.atheism	234	39	2	2
talk.religion.misc	49	159	2	3
comp.graphics	4	6	363	6
sci.space	2	2	4	383

TWSVM and LS-TWSVM. Tables 6 and 7 show confusion matrices for TWSVM and LS-TWSVM respectively. Here, it is observed that classification results have improved significantly with TWSVM and LS-TWSVM as compared to LS-SVM for 20 Newsgroups dataset. The non-diagonal entries are smaller in magnitude than the diagonal ones for TWSVM and LS-TWSVM. The accuracy achieved by the classifiers for 20 Newsgroups dataset is reported in Fig. 5. Here, LS-SVM outperforms the other two classifiers by achieving an accuracy of 92.96%.

Precision, Recall and *F*1 score for 20 Newsgroups dataset is reported for each document category in Table 8.

5.4 Training and testing time comparison

LS-TWSVM not only achieves the highest accuracy but it is much more time efficient than the other two classifiers.

Table 4 Reuters: precision, recall and *F*1 score

	LS-SVM			TWSVM			LS-TWSVM		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Acq	0.9356	0.7224	0.8153	0.9667	0.9544	0.9605	0.9897	0.9614	0.9754
Earn	0.9643	0.9563	0.9602	0.9736	0.9780	0.9758	0.9811	0.9909	0.9860
Money-fx	0.6287	1.0000	0.7720	0.9610	0.9899	0.9752	0.9707	1.0000	0.9851
Grain	0.2979	1.0000	0.4590	0.9541	0.9455	0.9498	0.9817	1.0000	0.9907
Average	0.7066	0.9197	0.7516	0.9638	0.9670	0.9653	**0.9808**	**0.9881**	**0.9843**

Table 7 LS-TWSVM 20 Newsgroups confusion matrix

Confusion matrix	alt.atheism	talk. religion. misc	comp.graph-ics	sci.space
alt.atheism	236	24	1	1
talk.religion.misc	26	173	2	3
comp.graphics	2	2	395	3
sci.space	0	6	15	371

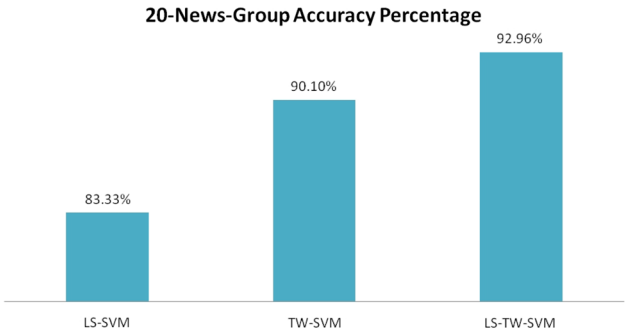


Fig. 5 The classification accuracy for 20 Newsgroups with all three classifiers

Since, LS-TWSVM solves a pair of systems of linear equations where each system is approximately half the size of the problem solved by LS-SVM. Therefore, LS-TWSVM is able to build the classifier model in lesser time than LS-SVM. The generalization ability of LS-TWSVM is much better than that of LS-SVM. TWSVM solves a pair of expensive QPPs, therefore LS-TWSVM is more efficient than TWSVM. The training-testing time of the three classifiers for Reuters dataset is shown graphically in Fig. 6. Since, there is huge difference in training-testing time of least-squares classifiers and TWSVM, so *y*-axis is taken in logarithmic scale. This clearly demonstrates the difference in learning times of the three classifiers. A similar trend is observed for 20 Newsgroups dataset in Fig. 6, where LS-TWSVM requires minimum time for building the classifier model whereas TWSVM takes te maximum time. This is due to the fact

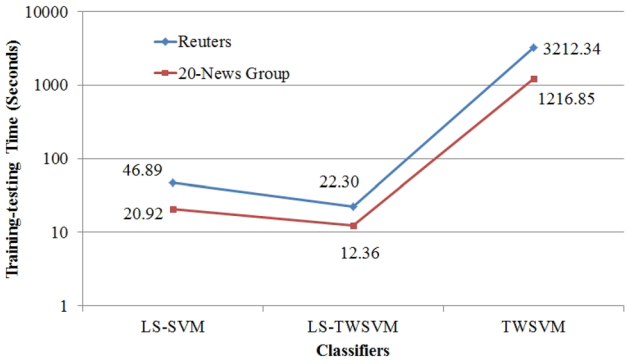


Fig. 6 Time in seconds utilized in training and testing for two datasets, with all three classifiers

that TWSVM solves a pair of expensive QPP along with two systems of linear equations. In contrast, least-squares version of classifiers avoids solving QPPs and generates the hyperplanes by solving only systems of linear equations. Table 9 presents the training-testing time (seconds) of three classifiers.

6 Conclusion

In this work, it is observed that Least Square Twin Support Vector Machine (LS-TWSVM) outperforms the other two variants of SVM. LS-TWSVM has better generalization ability and computational speed as compared to Least Square Support Vector Machine (LS-SVM) and Twin Support Vector Machine (TWSVM) for News Categorization problem. A significant increase in performance of computational time is achieved while using Least-squares version of classifiers. It is due to the fact that these classifiers solve systems of linear equations rather than solving quadratic programming problems.

The future line of work could be to investigate if further improvements can be achieved for other SVM based multi-category approaches. New SVM-based classifiers could be developed and experimentally inspected for similar applications. This would significantly enhance the superiority of SVM based approaches on Text Categorization problems.

Table 8 20 Newsgroup: precision, recall and *F*1 score

	LS-SVM			TWSVM			LS-TWSVM		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
alt.atheism	0.9444	0.5640	0.7062	0.8448	0.8097	0.8269	0.9008	0.8939	0.8973
talk.religion.mi	0.4654	0.8938	0.6121	0.7465	0.7718	0.7589	0.8480	0.8439	0.8460
comp.graphics	0.8980	0.9724	0.9337	0.9578	0.9784	0.9680	0.9826	0.9564	0.9693
sci.space	0.8997	0.9890	0.9423	0.9795	0.9721	0.9758	0.9464	0.9815	0.9636
Average	0.8019	0.8548	0.7986	0.8821	0.8830	0.8824	**0.9195**	**0.9189**	**0.9191**

Table 9 Training and testing time comparison matrix

Seconds	Reuters	20 Newsgroups
LS-SVM	46.89	20.92
TWSVM	3212.34	1216.85
LS-TWSVM	**22.30**	**12.36**

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. McCallum A, Nigam KA (1998) Comparison of event models for naive bayes text classification. In: AAAI-98 workshop on learning for text categorization, 26 Jul, vol 752, pp 41–48
2. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
3. Vapnik VN (2000) Methods of pattern recognition. In: The nature of statistical learning theory. Springer New York, pp 123–180
4. Vapnik VN (1999) An overview of statistical learning theory. IEEE Trans Neural Netw 10(5):988–999
5. Mangasarian OL, Musicant DR (2001) Lagrangian support vector training. J Mach Learn Res 1(Mar):161–77
6. Mangasarian OL, Wild EW (2001) Proximal support vector machine classifiers. In: Proceedings KDD-2001: knowledge discovery and data mining
7. Suykens JA, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Process Lett 9(3):293–300
8. Mangasarian OL, Wild EW (2005) Multisurface proximal support vector machine classification via generalized eigenvalues. IEEE Trans Pattern Anal Mach Intell 28(1):69–74
9. Jayadeva, Khemchandani R Chandra S (2007) Twin support vector machines for pattern classification. IEEE Trans Pattern Anal Mach Intell 29(5:905–910
10. Saigal P, Chandra S, Rastogi R (2019) Multi-category ternion support vector machine. Eng Appl Artif Intell 85:229–242
11. Saigal P (2017) Time efficient variants of twin support vector machine with applications in image processing. Ph.D. thesis, South Asian University, New Delhi
12. Peng X (2011) TPMSVM: a novel twin parametric-margin support vector machine for pattern recognition. Pattern Recognit 44(10):2678–2692
13. Khemchandani R, Saigal P, Chandra S (2016) Improvements on ν-twin support vector machine. Neural Netw 79:97–107
14. Rastogi R, Saigal P, Chandra S (2018) Angle-based twin parametric-margin support vector machine for pattern classification. Knowl Based Syst 139:64–77
15. Rastogi R, Saigal P (2017) Tree-based localized fuzzy twin support vector clustering with square loss function. Appl Intell 47(1):96–113
16. Ai Q, Wang A, Wang Y, Sun H (2018) Improvements on twin-hypersphere support vector machine using local density information. Prog Artif Intell 7(3):167–175
17. Kumar MA, Gopal M (2015) Least squares twin support vector training for text categorization. In: 39th National systems conference (NSC). IEEE, pp 1–5
18. Joachims T (1998) Text categorization with support vector training: Learning with many relevant features. In: European conference on machine learning, vol 21. Springer, Berlin, pp 137–142
19. Mitra V, Wang CJ, Banerjee S (2007) Text classification: a least square support vector machine approach. Appl Soft Comput 7(3):908–14
20. He J, Tan AH, Tan CL (2003) On machine learning methods for Chinese document categorization. Appl Intell 18(3):311–22
21. Lee LH, Wan CH, Rajkumar R, Isa D (2012) An enhanced support vector machine classification framework by using Euclidean distance function for text document categorization. Appl Intell 37(1):80–99
22. Mangasarian OL (1993) Nonlinear programming, vol 10. SIAM, Philadelphia
23. Kumar MA, Gopal M (2009) Least squares twin support vector machines for pattern classification. Expert Syst Appl 36(4):7535–7543
24. Bird S (2006) NLTK: the natural language toolkit. In: Proceedings of the COLING/ACL on interactive presentation sessions, 17 Jul 17. Association for Computational Linguistics, pp 69–72
25. Ramos J (2003) Using tf-idf to determine word relevance in document queries. In: Proceedings of the first instructional conference on machine learning
26. Hsu C-W, Lin C-J (2002) A comparison of methods for multiclass support vector machines. IEEE Trans Neural Netw 13(2):415–425
27. Duda RO, Hart PE, Stork DG (2012) Pattern classification. Wiley, New York