Pattern Recognition 43 (2010) 796–804

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Two-level k-means clustering algorithm for k–τ relationship establishment and linear-time classification

Radha Chitta [a,*], M. Narasimha Murty [b]

[a] Department of Electrical Engineering, Indian Institute of Science, Bangalore 560012, India
[b] Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India

ARTICLE INFO

Article history:
Received 12 January 2009
Received in revised form
25 August 2009
Accepted 22 September 2009

Keywords:
Clustering
k-Means
Classification
Linear-time complexity
Support vector machines
k-Nearest neighbor classifier

ABSTRACT

Partitional clustering algorithms, which partition the dataset into a pre-defined number of clusters, can be broadly classified into two types: algorithms which explicitly take the number of clusters as input and algorithms that take the expected size of a cluster as input. In this paper, we propose a variant of the k-means algorithm and prove that it is more efficient than standard k-means algorithms. An important contribution of this paper is the establishment of a relation between the number of clusters and the size of the clusters in a dataset through the analysis of our algorithm. We also demonstrate that the integration of this algorithm as a pre-processing step in classification algorithms reduces their running-time complexity.

1. Introduction

Clustering, the process of grouping similar patterns together, finds application in pattern analysis, decision making and various machine learning problems. Clustering algorithms [1] can be broadly classified into hierarchical and partitional clustering algorithms. Hierarchical algorithms, as the name suggests, produce a hierarchy of clusters. Agglomerative hierarchical clustering algorithms initially treat each pattern as a separate cluster and merge clusters recursively based on their distance from each other until a single cluster containing all the patterns is obtained. Divisive hierarchical algorithms start with a single cluster containing all the patterns and recursively divide the cluster till each cluster comprises a single pattern. Partitional clustering algorithms partition the dataset into a pre-defined number of clusters. They can further be classified into two categories based on their input parameters: algorithms which explicitly take the number of clusters k as input and algorithms that take as input, a threshold τ on the radius of the clusters which indirectly determines the number of clusters. k-means and k-medoids are examples of algorithms in the first category. Algorithms like Leader, DB-Scan, etc., fall in the second category.

In this paper, we propose the *two-level k-means clustering algorithm*, a variant of the MacQueens k-means algorithm. It takes τ as input and initially partitions the dataset into a predefined k' ($k' < k$) number of clusters. It then refines these clusters using the threshold τ to generate k clusters. We show both analytically and empirically that our algorithm requires lesser number of distance computations than standard k-means algorithms and hence gives better time performance. We establish a relationship between τ and k through the analysis of our algorithm which forms a bridge between the two types of partitional clustering algorithms. We also establish a bound on the clustering error of our algorithm.

We finally present a generic scheme for the integration of the two-level k-means algorithm into classification algorithms. We prove that this integration renders the running time complexity of the classification algorithm linear. We integrate our algorithm in Support Vector Machines and k-NN classifier and thus demonstrate that better time performance is achieved.

2. Background

In this section, we present a brief introduction to the k-means problem and associated k-means clustering algorithms that the proposed algorithm is based on.

* Corresponding author. Present address: Yahoo! Software Development India, Torrey Pines, Embassy Golf Links Business Park, Off Indiranagar - Koramangala Intermediate Ring Road, Bangalore 560071, India. Tel.: +91 80 30774396; fax: +91 80 30774455.
E-mail addresses: cradha@ymail.com (R. Chitta), mnm@csa.iisc.ernet.in (M. Narasimha Murty).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.09.019

2.1. k-Means algorithm

The k-means problem is to determine k points called *centers* so as to minimize the *clustering error*, defined as the sum of the distances of all data points to their respective cluster centers. The most commonly used algorithm for solving this problem is the *Lloyd's k-means algorithm* [2,3] which iteratively assigns the patterns to clusters and computes the cluster centers.

MacQueens k-means algorithm [4] is a two-pass variant of the Lloyd's k-means algorithm:

1. Choose the first k patterns as the initial k centers. Assign each of the remaining $N-k$ patterns to the cluster whose center is closest. Calculate the new centers of the clusters obtained.
2. Assign each of the N patterns to one of the k clusters obtained in step 1 based on its distance from the cluster centers and recompute the centers.

Analysis: Distance computation is the only time-consuming operation in this algorithm. So, we focus on the number of distance computations performed.

In step 1, the number of distance computations needed is given by $k(N-k)$. The number of distance computations in step 2 equals Nk. This implies that the total number of distance computations equals $k(N-k)+Nk=2Nk-k^2$ and the complexity is $O(Nk)$.

3. Related work

Our two-level k-means algorithm belongs to the class of multi-level clustering algorithms. These techniques essentially involve the following generic steps:

- divide the given dataset into blocks,
- cluster each block separately into a predefined number of clusters, and
- cluster the cluster centers obtained from all the blocks into the required number of clusters.

Multi-level clustering enables easier handling of large datasets since only one block needs to be held in the memory at any time. The divide-and-conquer strategy employed allows parallelization which, in turn, increases the clustering speed. Also, different clustering algorithms can be adopted at different levels allowing the incorporation of domain knowledge.

Algorithm Small-Space [5] arbitrarily partitions the input dataset χ into l blocks. Each block is clustered into k clusters. A second level dataset χ' is formed from the $O(lk)$ cluster centers, each weighted by the number of patterns assigned to it. χ' is then clustered into k clusters. This can be extended to multiple levels. It has been proved that the clustering error of the solution obtained through *Algorithm Small-Space* is a constant-factor approximation of the clustering error of the optimal clustering algorithm.

Clustering algorithms have been employed to improve the efficiency of different classifiers. We apply our clustering-based classification technique to Support Vector Machines and the k-NNC classifier.

Clustering-based SVM (CB-SVM) [6] integrates BIRCH [7], a hierarchical micro-clustering algorithm into SVM. CB-SVM reduces the number of training patterns that are input to SVM by first clustering the dataset using BIRCH and identifying the patterns that will contribute to the learning process. Empirical results show that CB-SVM achieves a significant reduction in the training time and improvement in test accuracy.

Zhang and Srihari [8] construct a Cluster Tree for the given data. The tree is traversed top-down based on the similarity between the test pattern and the nodes at each level to determine the k -nearest neighbors of the test pattern. The class of the test pattern is then determined by majority voting.

4. Two-level *k*-means clustering algorithm

In this section, we present the *two-level k-means clustering algorithm* (see Fig. 1). We prove that it requires lesser number of distance computations than MacQueens k-means algorithm.

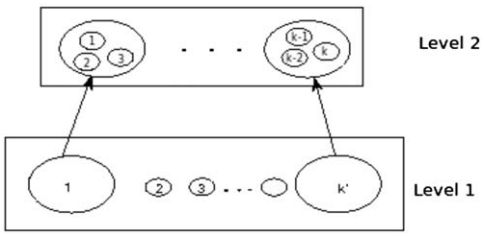

Fig. 1. Two-level k-means clustering algorithm.

Further analysis establishes the relationship between the number of clusters in the dataset and the expected size of each cluster. We then prove that its clustering error is less than twice the error of the optimal clustering algorithm.

Inputs: Dataset $\chi=\{x_i\}_{i=1}^{N}$, $x_i\in\Re^n$; Radius threshold τ
Algorithm:

1. In the first level, cluster the given dataset into an arbitrarily chosen k' number of clusters.
2. Calculate the radius r_i $(i=1,2,\ldots,k')$ of the ith cluster.
3. If the radius of the cluster is greater than the user-defined threshold τ, split it using k-means with the number of clusters set to $(r_i/\tau)^n$. This is the second level of clustering which yields a total of k clusters, each having a radius less than τ.

4.1. Analysis

4.1.1. Bound on number of distance computations

We are given a dataset $\chi=\{x_1,x_2,\ldots,x_N\}$ where $x_i\in\Re^n$ are independent samples, all drawn from the same distribution (see Table 1).

On using the MacQueens two-pass k-means algorithm for clustering the data into k clusters, the number of distance computations made is given by

$$ND_1=2Nk-k^2 \tag{4.1}$$

Suppose we use the same two-pass algorithm in both the levels of the two-level k-means algorithm described. Then the number of distance computations in level 1 is given by

$$ND_{L1}=2Nk'-(k')^2$$

After the first level of clustering, we have k' clusters $\{\chi_1,\chi_2,\ldots,\chi_{k'}\}$ with centers $\{c_1,c_2,\ldots,c_{k'}\}$. Let us define the radius of a cluster χ_i as

$$r_i=\max_{x_j\in\chi_i} d(x_j,c_i) \tag{4.2}$$

where $d(x,y)$ is the metric distance between vectors x and y. Cluster i is sub-clustered in the second level if $r_i>\tau$. For $1\le i\le k'$

Table 1
List of notations.

Notation	Definition
χ	Training dataset
x_i	Pattern i
n	Dimensionality of the dataset
N	Number of training patterns
τ	Radius threshold
k'	Number of clusters in the first level
k	Final number of clusters
R	Radius of the ball enclosing all the training patterns
c_i	ith cluster center
r_i	Radius of the ith cluster
w_i	Number of patterns assigned to the ith cluster
p_i	Probability that radius of the ith cluster is greater than τ
$E[r_i]$	Expected value of r_i
P_1, P_2	Lower and upper bounds on p_i
β, α	Lower and upper bounds on r_i/τ
ND_1	Number of distance computations required to cluster N patterns into k clusters using MacQueens algorithm
ND_{L1}	Number of distance computations performed in the first level
ND_{L2}	Number of distance computations performed in the second level
ND_2	Number of distance computations performed by the two-level k-means algorithm
C	Constant defined as $C=(k-k')(2N-k-k')$

define indicator variable

$$Z_i = 1_{\{r_i > \tau\}} \tag{4.3}$$

and probability

$$p_i = P[Z_i = 1] \tag{4.4}$$

Let w_i be the number of patterns in cluster χ_i. If a cluster χ_i moves to second level, we sub-cluster χ_i into $(r_i/\tau)^n$ clusters. Therefore, the number of distance computations in cluster χ_i is given by

$$ND^i_{L2} = 2w_i\left(\frac{r_i}{\tau}\right)^n - \left(\frac{r_i}{\tau}\right)^{2n}$$

The total number of distance computations in level 2 is given by

$$ND_{L2} = \sum_{i=1}^{k'} Z_i ND^i_{L2} = \sum_{i=1}^{k'} Z_i\left(2w_i\left(\frac{r_i}{\tau}\right)^n - \left(\frac{r_i}{\tau}\right)^{2n}\right) \tag{4.5}$$

Adding ND_{L1} and ND_{L2} and taking expectation, we obtain the expected number of distance computations ND_2 performed by the two-level k-means algorithm:

$$ND_2 = 2Nk' - (k')^2 + \sum_{i=1}^{k'} p_i\left(2w_i\left(\frac{r_i}{\tau}\right)^n - \left(\frac{r_i}{\tau}\right)^{2n}\right) \tag{4.6}$$

So,

$$ND_1 - ND_2 = 2Nk - k^2 - 2Nk' + (k')^2 + \frac{1}{\tau^{2n}}\sum_{i=1}^{k'} p_i r_i^{2n} - \frac{2}{\tau^n}\sum_{i=1}^{k'} w_i p_i r_i^n \tag{4.7}$$

Let us assume the following bounds for p_i and r_i:

$$0 \le P_1 \le p_i \le P_2 \le 1 \quad \forall i \tag{4.8}$$

$$\beta\tau \le r_i \le \alpha\tau \quad \forall i, \quad \alpha \ge 0 \text{ and } \beta \ge 0 \tag{4.9}$$

Using the above bounds, we obtain

$$ND_1 - ND_2 \ge (k-k')(2N-k-k') + \frac{1}{\tau^{2n}}\sum_{i=1}^{k'} P_1\beta^{2n}\tau^{2n} - \frac{2}{\tau^n}\sum_{i=1}^{k'} w_i P_2\alpha^n\tau^n \ge (k-k')(2N-k-k') - 2NP_2\alpha^n \tag{4.10}$$

Similarly,

$$ND_1 - ND_2 \le (k-k')(2N-k-k') + \frac{1}{\tau^{2n}}\sum_{i=1}^{k'} P_2\alpha^{2n}\tau^{2n} - \frac{2}{\tau^n}\sum_{i=1}^{k'} w_i P_1\beta^n\tau^n \le (k-k')(2N-k-k') + k'P_2\alpha^{2n} \tag{4.11}$$

The radii values are non-negative random variables. Therefore, by Markov inequality,

$$P[r_i \ge \tau] \le \frac{E[r_i]}{\tau}$$

It can be proved that $P_2 \le E[r_i]/\tau$. Using this inequality and taking $C=(k-k')(2N-k-k')$, we obtain

$$C - \frac{2NE[r_i]\alpha^n}{\tau} \le ND_1 - ND_2 \le C + \frac{k'E[r_i]\alpha^{2n}}{\tau} \tag{4.12}$$

For example, let us assume that $r_i \sim U(0,R)$ where R is the radius of the ball enclosing all the training examples. Then, $E[r_i]=R/2$. Substituting this value in relation (4.12), we obtain

$$C - \frac{N\alpha^n R}{\tau} \le ND_1 - ND_2 \le C + \frac{k'\alpha^{2n}R}{2\tau} \tag{4.13}$$

The following lemma establishes a bound on the number of distance computations performed in the second level:

Lemma 4.1. *When the value of k' is chosen such that $k' \ll k$, the expected number of distance computations in the second level is bounded above by $N\alpha^n R/\tau$.*

Proof. From relation (4.13) we have $ND_1 - ND_2 \ge (k-k')(2N-k-k') - N\alpha^n R/\tau$. When $k' \ll k$, $(k-k')(2N-k-k') \sim k(2N-k) = ND_1$. Therefore $ND_2 \le N\alpha^n R/\tau$. □

4.1.2. Values of the parameters

As the value chosen for k' tends to k (the final number of clusters obtained), the difference in the number of distance computations tends to 0. So, to obtain a considerable difference in the number of distance computations, k' should be chosen to satisfy $k' \ll k$. The value of k' can be chosen as some small value between 2 and 10. Further, domain knowledge can be incorporated while choosing this value.

Lemma 4.2. *When k' is equal to k, the difference in the number of distance computations $ND_1 - ND_2$ is zero.*

Proof. At the end of the clustering process, we obtain k clusters each cluster satisfying the condition that its radius is not greater than τ. If k' is chosen equal to k, all the first level clusters will have a radius less than τ. Since no clustering is performed in the second level, we have $ND_1 = ND_2$. □

The value of τ determines the number of clusters that move to the second level for sub-clustering, which in turn, determines the number of distance computations required to cluster the given dataset. To ensure that the number of distance computations made by the two-level k-means algorithm is lesser than or equal to the number of distance computations performed by MacQueens k-means on the entire dataset, we choose τ as follows:

We require $ND_1 - ND_2 \ge 0$. It is sufficient if we ensure that the lower bound on $ND_1 - ND_2$ is greater than or equal to zero. From Eq. (4.13), we have

$$ND_1 - ND_2 \ge C - \frac{N\alpha^n R}{\tau}$$

We must ensure

$$C - \frac{N\alpha^n R}{\tau} \ge 0$$

$$\Rightarrow \tau \geq \frac{N\alpha^n R}{C}$$

Since the radii values are bounded by R, τ cannot be chosen to be greater than R. Hence, we have the following inequality for choosing the value of the threshold parameter τ:

$$\frac{N\alpha^n R}{C} \leq \tau \leq R \tag{4.14}$$

4.1.3. Relation between k and τ

Let us consider the case when $k'=1$, i.e. we directly cluster the dataset into $(R/\tau)^n$ clusters. Using this scenario, we can obtain a relationship between the radius threshold τ and the final number of clusters k. The number of distance computations required in this scenario should not be greater than the number of computations required to cluster the dataset into k clusters. Therefore, we have

$$2Nk - k^2 \geq 2N\left(\frac{R}{\tau}\right)^n - \left(\frac{R}{\tau}\right)^{2n}$$

$$\Rightarrow \left(\left(\frac{R}{\tau}\right)^n - k\right)\left(\left(\frac{R}{\tau}\right)^n + k - 2N\right) \geq 0$$

Since the maximum values that $(R/\tau)^n$ and k can take are N, the expression $((R/\tau)^n + k - 2N)$ cannot be positive. Hence, the following two inequalities must be satisfied:

$$\left(\frac{R}{\tau}\right)^n \leq k \quad \text{and} \quad \left(\frac{R}{\tau}\right)^n \leq (2N-k)$$

$$\Rightarrow \tau \geq \frac{R}{(k)^{1/n}} \quad \text{and} \quad \tau \geq \frac{R}{(2N-k)^{1/n}}$$

This gives us the following relation between τ and k:

$$\max\left(\frac{R}{(k)^{1/n}}, \frac{R}{(2N-k)^{1/n}}\right) \leq \tau \leq R \tag{4.15}$$

4.1.4. Bound on clustering error

Let $c_1^*, c_2^*, \ldots, c_k^*$ be the centers corresponding to the optimal clustering of the data into k-clusters. Let indicator variable $Z_{li}^* = 1$ if pattern x_l belongs to the cluster with center c_i^* (see Table 2). The error of the optimal clustering is given by

$$E_{opt} = \sum_{l=1}^{N}\sum_{i=1}^{k} Z_{li}^* d(x_l, c_i^*) \tag{4.16}$$

Let the centers obtained through the first level of our algorithm be $c_1, c_2, \ldots, c_{k'}$. On splitting the first-level clusters in the second level, we obtain centers $c_{1'1}, c_{1'2}, \ldots, c_{1(r_1/\tau)^{n}}, \ldots, c_{k'1}, c_{k'2}, \ldots, c_{k'(r_{k'}/\tau)^{n}}$.

For the first level clusters whose radius is less than τ, $c_{ij}' = c_i \ \forall j$.

Table 2
List of notations.

Notation	Definition
c_i^*	Optimal cluster centers
c_i	Cluster centers obtained in the first level of clustering
c_{ij}'	Cluster centers obtained in the second level of clustering
Z_{li}^*	Indicator variable, true if pattern x_l belongs to i th optimal cluster
Z_{li}	Indicator variable, true if pattern x_l belongs to i th first-level cluster
Z_{lij}	Indicator variable, true if pattern x_l belongs to j th sub-cluster of the i th first level cluster
$d(a,b)$	Metric distance from point a to point b
E_{opt}	Optimal clustering error
E	Clustering error of the two-level k-means algorithm

Let indicator variables $Z_{li}=1$ if pattern x_l belongs to i th first level cluster and $Z_{lij}'=1$ if pattern x_l belongs to j th sub-cluster of i th first level cluster. The final error E is given by

$$E = \sum_{l=1}^{N}\sum_{i=1}^{k'}\sum_{j=1}^{(r_i/\tau)^n} Z_{lij}'\, d(x_l, c_{ij}') \tag{4.17}$$

The second level clusters are obtained by splitting some of the first level clusters and hence the error reduces from the first level to the second level, i.e.

$$E \leq \sum_{l=1}^{N}\sum_{i=1}^{k'} Z_{li} d(x_l, c_i) \tag{4.18}$$

For any x_l whose closest centers are c_i and c_j^* in the first-level and the optimal clusterings respectively, by triangle inequality

$$d(x_l, c_i) \leq d(x_l, c_j^*) + d(c_i, c_j^*)$$

Summing over all the training patterns,

$$\sum_{l=1}^{N}\sum_{i=1}^{k'} Z_{li} d(x_l, c_i) \leq \sum_{l=1}^{N}\sum_{j=1}^{k} Z_{lj}^* d(x_l, c_j^*)$$

$$+ \sum_{l=1}^{N} Z_{li} Z_{lj}^* d(c_i, c_j^*) \leq \sum_{l=1}^{N}\sum_{j=1}^{k} Z_{lj}^* d(x_l, c_j^*) + \sum_{l=1}^{N}\sum_{j=1}^{k} Z_{lj}^* d(x_l, c_j^*)$$

Using relation (4.18), we have

$$E \leq 2E_{opt} \tag{4.19}$$

5. Application of two-level k-means for classification

The proposed two-level k-means algorithm can be employed to reduce the time complexity of classification algorithms. A generic algorithm for the integration of the clustering technique into classification algorithms is described below:

Inputs: Dataset $\chi = \{x_i, y_i\}_{i=1}^N$, $x_i \in \Re^n$, $y_i \in \Upsilon$; Initial number of clusters k; Radius threshold τ

Algorithm:

1. Cluster the dataset χ into k clusters and determine the radius of each cluster.
2. Train the classifier using the centers of those clusters which have their radius greater than τ. Noisy patterns in the data generally form small-sized clusters. Elimination of clusters with small radii reduces the effect of outliers.
3. Determine the clusters which form a part of the classification model. Sub-cluster these clusters and use these centers to train the classifier.
4. Again determine the clusters in the classification model and train the classifier with the patterns in these clusters.

All the N patterns are processed only in step 1. In step 2 only $O(k)$ patterns are processed. In steps 3 and 4, the number of patterns processed is much lesser than N, when N is large. A significant number of patterns are eliminated either because they belong to clusters which have a small radius or because they are not part of the model. Hence, the first step, which has an $O(N)$ time complexity for constant k, alone determines the time complexity of the algorithm. Hence the training time complexity of the classifier integrated with two-level k-means algorithm becomes linear.

5.1. Application of two-level k-means in SVM

Training an SVM [9,10] involves solving a quadratic programming (QP) problem. The training time complexity of QP is $O(N^3)$

and its space complexity is at least quadratic. When the training dataset is large in size, standard QP solvers fail to scale up.

Decomposition methods such as Chunking [11] and Sequential Minimal Optimization (SMO) [12] choose a set of Lagrange variables to optimize in each iteration and solve the optimization problem involving these variables. Core Vector Machines (CVM) [13] formulate the SVM problem as a minimum enclosing ball (MEB) problem and obtain an approximate solution with significant speedup. Cutting-Plane Algorithm [14] is used to solve an alternative but equivalent formulation of the SVM optimization problem called the Structural SVM. In each iteration, it considers a few violated constraints and finds the solution which satisfies these constraints. This is continued till a solution is obtained to the desired approximation. The implementation of this algorithm is called SVM^{perf} and it has linear time complexity.

Clustering-based SVM (CB-SVM) [6] integrates a scalable hierarchical micro-clustering algorithm, BIRCH [7], into SVM to reduce the number of patterns that are processed by the learner. Since lesser number of patterns are processed, the time taken for training reduces significantly.

We integrate the two-level k-means algorithm in the SVM training process to realize a linear time SVM as follows:

Inputs: Dataset $\chi = \{x_i, y_i\}_{i=1}^{N}$, $x_i \in \Re^n$, $y_i \in \{-1, 1\}$; Initial number of clusters k; Radius threshold τ

Algorithm:

1. Cluster the positive and negative patterns independently and calculate the radius of each cluster obtained.
2. Train an SVM boundary function from the centers of the clusters whose radius is greater than the user-defined threshold. Noisy patterns generally form small-sized clusters and hence these clusters can be ignored.
3. Subcluster the clusters which are near the boundary.
4. Construct another SVM from the centers which are obtained in the previous step.
5. Decluster the clusters near the boundary of the SVM constructed in step 4 and train an SVM on the patterns in these clusters. This gives the final SVM boundary function.

In step 3, we need to determine the clusters which are close to the boundary. *Closeness to the boundary* [6] is defined as follows: Let D_i be the distance of a cluster center to the boundary and R_i be the radius of the cluster. Let D_s be the maximum of the distances from the support vector centers to the boundary. A cluster is said to be near the boundary if

$$D_i - R_i < D_s \tag{5.1}$$

5.1.1. Time complexity

The first step of the algorithm involves clustering the given patterns into k clusters. Using the MacQueens k-means algorithm, we can obtain the clustering in $O(Nk)$ time.

Assuming that sequential minimal optimization (SMO) is employed for training the SVM, the training time complexity with N patterns would be $O(N^2)$. SVM training is performed in steps 2, 4 and 5.

In step 2, the number of training patterns equals the number of clusters which have a radius greater than the threshold. This is equal to at most k clusters from the positive patterns plus at most k clusters from the negative patterns. So the training complexity is $O((2k)^2) \sim O(k^2)$.

In step 3, the number of patterns which are clustered equal

$$N_1 = \sum_{i=1}^{k} w_i Y_i$$

where w_i is the number of patterns in the ith cluster and

$$Y_i = 1_{\{i\text{th cluster is close to the boundary}\}}$$

Hence, the time complexity of this step is $O(N_1 k^2)$. The number of patterns input to the SVM in step 4 is at most k^2 and hence the time complexity of this step is $O(k^4)$.

The number of patterns input to the SVM constructed in step 5 is

$$N_2 = \sum_{i=1}^{k} \sum_{j=1}^{k} Y'_{ij}\, w'_{ij}$$

where w'_{ij} is the number of patterns in the j th subcluster of the i th cluster and

$$Y'_{ij} = 1_{\{Y_i = 1 \text{ and the } j\text{th subcluster is close to the boundary}\}}$$

The training complexity of the final SVM obtained from step 5 is $O(N_2^2)$.

The total time complexity is hence given by $O(Nk + k^2 + N_1 k^2 + k^4 + N_2^2)$. When the dataset is large in size, we will have $N_1 \ll N$ and $N_2 \ll N$. For such large datasets, the time complexity becomes $O(Nk) \sim O(N)$. Hence for large datasets, SVM integrated with two-level k-means algorithm has linear time complexity.

5.1.2. Space complexity

At any point in time, it is required to store only the input patterns and the k cluster centers. Hence the space complexity is $O(N + k) \sim O(N)$.

5.2. Application of two-level k-means in k-NNC

k-NNC algorithm becomes computationally intensive when the size of the training set is large. Clustering is used to realize an efficient k-NNC variant in [8]. This technique achieves considerable reduction in computation but its time complexity is non-linear. We prove that, on incorporating two-level k-means algorithm into the k-NN classifier, the time complexity becomes linear.

The two-level k-means algorithm can be incorporated into k-NN classification as follows:

Inputs: Training Dataset $\chi = \{x_i, y_i\}_{i=1}^{N}$, $x_i \in \Re^n$, $y_i \in \Upsilon$; Test Pattern TP; Initial number of clusters k'; Radius threshold τ; Number of neighbors k

Algorithm:

1. Cluster the dataset χ into k' clusters and determine the radius of each cluster.
2. Find k cluster centers nearest to TP from those centers whose clusters have radius greater than τ.
3. Subcluster the k clusters obtained in the previous step and find the k nearest sub-cluster centers.
4. From the patterns in the nearest sub-clusters, find the k nearest patterns. Assign TP the class to which majority of these patterns belong.

5.2.1. Time complexity

The first step of the algorithm involves clustering the given patterns into k' clusters. Using the MacQueens k-means algorithm, we can obtain the clustering in $O(Nk')$ time.

In step 2, $O(k')$ distances need to be computed. Sorting these distances and finding the k nearest centers would have a complexity of $O((k')^2)$ even if a simple sorting algorithm such as bubble sort is employed.

In step 3, the number of patterns which are clustered equal

$$N_1 = \sum_{i=1}^{k} w_i Y_i$$

where w_i is the number of patterns in the i th cluster and

$$Y_i = 1_{\{i\text{th center is among the } k\text{−NN}\}}$$

To find the k nearest sub-cluster centers, $O((kk')^2)$ effort is required. Hence, the time complexity of this step is $O(N_1 kk' + (kk')^2)$.

The number of patterns input to step 4 is

$$N_2 = \sum_{i=1}^{k} \sum_{j=1}^{k} Y'_{ij}\, w'_{ij}$$

where w'_{ij} is the number of patterns in the j th subcluster of the i th cluster and

$$Y'_{ij} = 1_{\{Y_i = 1 \text{ and the } j\text{th sub−cluster center is among the } k\text{−}NN\}}$$

Finding the k-nearest neighbors among these patterns requires the sorting of N_2 distances which requires $O(N_2^2)$ effort.

The total time complexity is hence given by $O(Nk' + (k')^2 + N_1 kk' + (kk')^2 + N_2^2)$. When the dataset is large in size, we will have $N_1 \ll N$ and $N_2 \ll N$. Hence, the time complexity becomes $O(N)$ for constant k and k'.

5.2.2. Space complexity

At any point in time, storage is required only for the input patterns, the k' cluster centers and the k nearest neighbors. Hence the space complexity is $O(N+k+k') \sim O(N)$.

6. Experimental results

6.1. Two-level k-means algorithm

The performance of two-level k-means algorithm has been tested on a synthetic dataset consisting of 1000 two-dimensional patterns. The dataset is generated as follows:

1. Select k cluster centers $\{c_i\}_{i=1}^{k}$ and k values for the radii $\{r_i\}_{i=1}^{k}$ of the clusters uniformly at random between 0 and R.
2. For each cluster C_i, sample l points uniformly around its center c_i and lying within a distance r_i from c_i.

We chose the parameter values $k=10$, $R=15$ and $l=100$ to generate the dataset (Fig. 2). The number of distance computations saved $(ND_1 - ND_2)$ by employing this algorithm (with $k'=2$) and the theoretical upper and lower bounds on this value are illustrated in Fig. 3. It is observed that a significant saving in the number of computations is made when the value of radius threshold parameter is chosen beyond the value 10. This shows that when the values of τ and k' are chosen to satisfy the constraints in Section 4.1.2, our algorithm is faster than standard k-means algorithm.

The relation between the threshold parameter τ and the number of clusters k was derived in Section 4.1.3. To demonstrate this experimentally, we find the maximum of the values $R/(k)^{1/n}$ and $R/(2N-k)^{1/n}$. Varying the radius threshold from this value to R we run k-means with the number of clusters set to $(R/\tau)^n$ and record the number of distance computations performed. We then run k-means with number of clusters set to the true value 10. The number of distance computations performed in both the cases are plotted in Fig. 4. A significant difference in the number of distance computations is observed, proving the validity of relation (4.15).

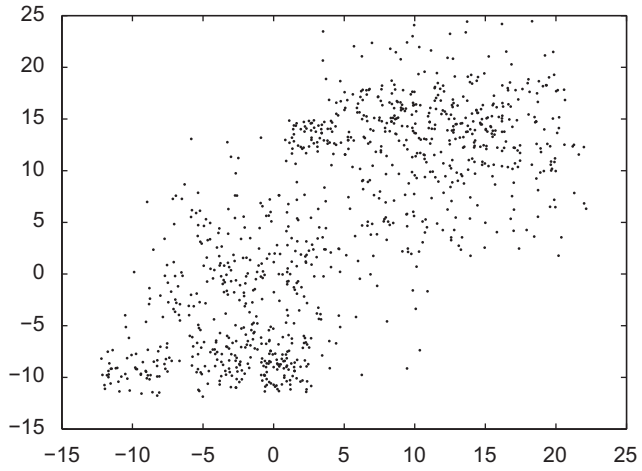

Fig. 2. Dataset used for comparing standard k-means algorithm with two-level k-means algorithm.

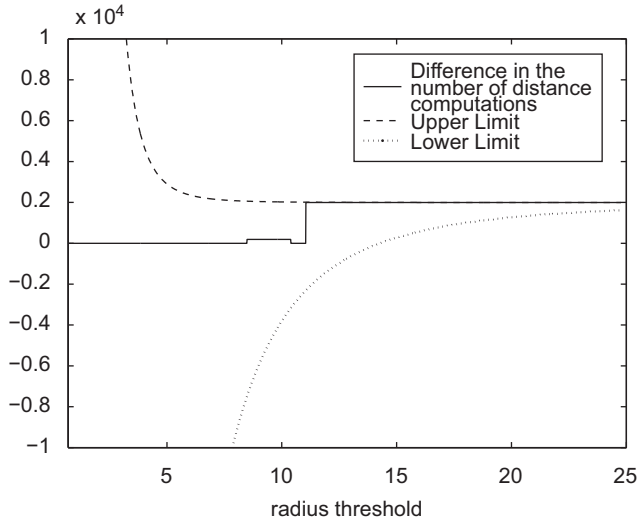

Fig. 3. Plot of $ND_1 - ND_2$ along with its upper and lower limits.

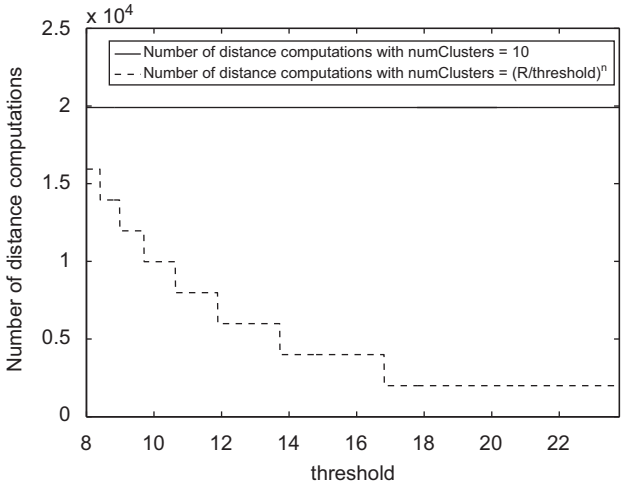

Fig. 4. Number of distance computations.

6.2. Two-level k-means in SVM

Integration of two-level k-means into SVM was tested on both synthetic and real datasets. It has been observed that there is considerable reduction in the training and testing time and the accuracy of the classifier is comparable to the accuracy of the SVM trained on the entire dataset at once. The testing was performed on an Intel(R) Xeon(R) 2 GHz machine with 4096 KB cache and 4038 MB memory.

6.2.1. Synthetic datasets

To verify that there is a substantial decrease in the training and testing time when two-level k-means clustering is incorporated in SVM training, we tested the algorithm on two synthetic datasets:

Dataset 1: This dataset contains a total of 100,000 two-dimensional patterns. The patterns of each class are drawn from independent normal distributions with means $[1\ 5]^T$ and $[10\ 5]^T$ and unit variance. The two classes are linearly separable. The test dataset, consisting of 80,000 patterns, is drawn from the same distribution. The results are presented in Tables 3 and 4. The first column in Table 3 shows the results corresponding to the SVM^{light} $V6.01$ implementation of SVM trained on the entire dataset without performing clustering (*Plain SVM*). The remaining columns in this table are the results for SVM^{light} with two-level k-means incorporated. The first column in Table 4 shows the results corresponding to the SVM^{perf} $V2.10$ implementation of SVM trained on the entire dataset without performing clustering. The remaining columns in this table are the results for SVM^{perf} with two-level k-means incorporated. We record the time taken for training and testing (in seconds), the test accuracy achieved and the number of support vectors for varying values of the parameters NumClusters(k) and Threshold(τ). The training time reduces by 28% and the number of support vectors reduce by 82% (with parameters k and τ chosen as 4 and 7 units, respectively) when compared to SVM^{light}. Hence, SVM integrated with two-level k-means performs better than chunking or sequential minimal optimization and is on par with the cutting plane algorithm for linearly separable data.

Dataset 2: This dataset contains a total of 1,000,000 two-dimensional patterns. The patterns of each class are drawn from independent normal distributions with means $[1\ 5]^T$ and $[5\ 5]^T$ and unit variance. This dataset is not linearly separable and hence is more realistic. The test dataset, consisting of 600,000 patterns, is drawn from the same distribution. The time taken for training and testing (in seconds), the test accuracy achieved and the number of support vectors for varying values of the parameters NumClusters(k) and Threshold(τ) are recorded in Table 5. The first column in the table shows the results corresponding to plain SVM. The remaining columns are the results for SVM (SVM^{light}) with two-level k-means incorporated. Here, we observe a heavy reduction in the training time and complexity. With parameters k and τ chosen as 10 and 3 units, respectively, the training time reduces by 48% and the number of support vectors reduce by 45% maintaining a test accuracy of 97%. Also, with minimal reduction in test accuracy (94.97%), the training time reduces to 12.74 s. It can be inferred from the above results that incorporating two-level k-means in SVM training greatly reduces the running time of SVM, especially when the training set is large in size.

6.2.2. Real dataset

We present the results obtained from experiments performed on different class combinations of the Optical Character Recognition (OCR) dataset. This dataset consists of handwritten characters representing the numerals 0–9. The training set consists of 667 patterns and the test set consists of 333 patterns for each class. Each pattern has 192 features and a class label which ranges from 0 to 9. For the purpose of experimentation with SVM, we consider combinations of classes, 1 vs 6 and 3 vs 8.

For each of the two class combinations, we record the test accuracy and the number of support vectors. The number of support vectors is an indicator of the complexity of the trained

Table 3
Training and testing timings for synthetic dataset 1 (using SVM^{light}).

	SVM^{light}	SVM with two-level k-means			
NumClusters	–	5	4	3	4
Threshold	–	3	5	10	7
Training time	**1.01**	1.01	1.00	0.69	**0.72**
Testing time	0.04	0.04	0.03	0.02	0.03
Accuracy	**100**	100	100	99.94	**99.62**
NumSVs	**57**	54	49	15	**10**

Table 4
Training and testing timings for synthetic dataset 1 (using SVM^{perf}).

	SVM^{perf}	SVM with two-level k-means	
NumClusters	–	10	10
Threshold	–	5	5.5
Training time	0.94	0.94	0.91
Testing time	0.01	0.02	0.01
Accuracy	99.98	99.23	99.67
NumSVs	2	2	2

Table 5
Training and testing timings for synthetic dataset 2.

	Plain SVM	SVM with two-level k-means		
NumClusters	–	10	7	10
Threshold	–	3	5	6
Training time	**5287.98**	**2731.46**	1037.95	**12.74**
Testing time	0.26	0.34	0.30	0.26
Accuracy	**97.71**	**97.04**	96.32	**94.97**
NumSVs	**57,894**	**31,825**	37,060	**1209**

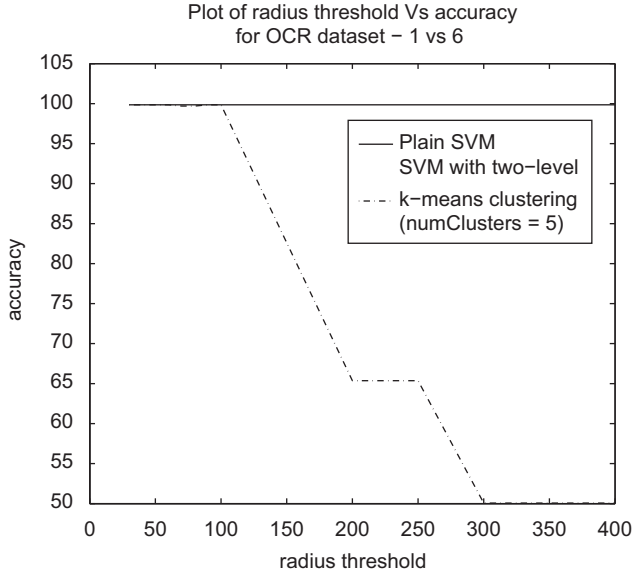

Fig. 5. Results for OCR 1 vs 6—test accuracy.

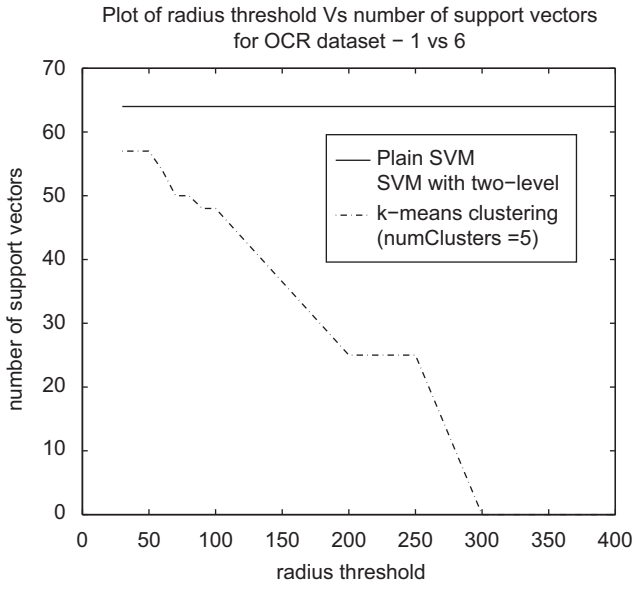

Fig. 6. Results for OCR 1 vs 6—number of support vectors.

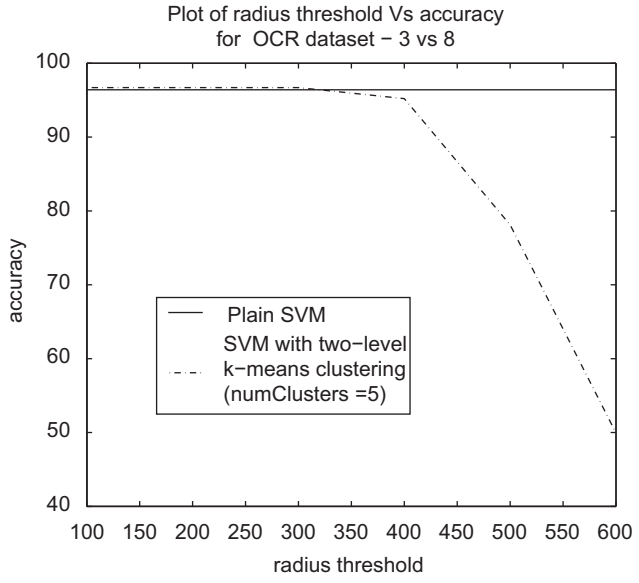

Fig. 7. Results for OCR 3 vs 8—test accuracy.

classifier and is also a bound on the probability of error [10]. Hence, up to a certain lower limit, lower the number of support vectors, better is the classifier. If the number of support vectors reduces beyond this lower limit, the classifier's performance deteriorates.

The results are presented in Figs. 5–8. In these figures, we observe that the number of support vectors after clustering is considerably lesser than the number of support vectors with plain SVM with a comparable test accuracy. For the class combination 1 vs 6 (Figs. 5 and 6), with a radius threshold of 90 units, the reduction in the number of support vectors is approximately 25% with no reduction in accuracy. For the class combination 3 vs 8 (Figs. 7 and 8), with a radius threshold of 300 units, there is an improvement in the test accuracy with a reduction of 27% in the number of support vectors. This can be attributed to the fact that some of the patterns which cause SVM to overfit the training data are eliminated during the cluster elimination process. For both the class combinations, the accuracy reduces as the threshold increases beyond a certain limit.

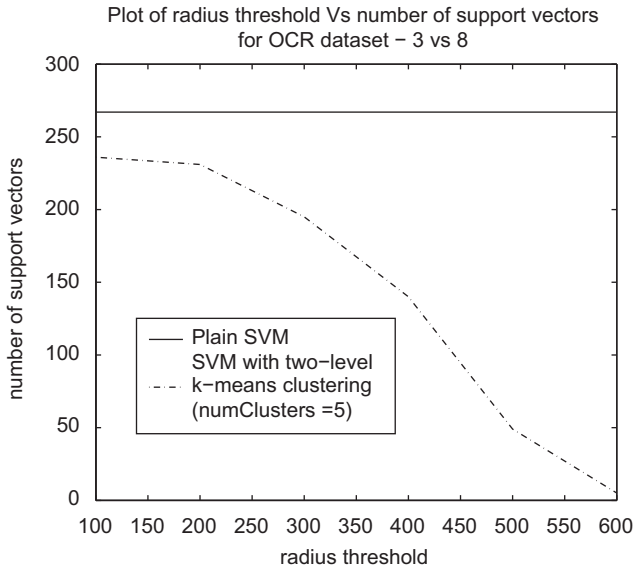

Fig. 8. Results for OCR 3 vs 8—number of support vectors.

Table 6
Comparison with CB-SVM.

	CB-SVM	SVM with two-level k-means		
Training set size	113,601	120,738	120,738	120,738
NumClusters	–	10	17	20
Threshold	–	0.05	0.05	0.05
Training time	10.589	14.09	4.62	2.91
Accuracy	99.00	99.00	96.00	95.00

This is due to the loss of important training examples from the training set, in turn, resulting in the reduction of the number of support vectors beyond the lower limit.

6.2.3. Comparison with other techniques

In order to demonstrate that our algorithm is on par with the methods that are currently employed to reduce the training time of SVM, we performed empirical comparisons with CB-SVM.

To compare with CB-SVM, we use the synthetic dataset described in [6]:

1. Randomly create k clusters with centers randomly chosen in the range $[c_l, c_h]$ for each dimension independently, radii in the range of $[r_l, r_h]$ and the number of points in each cluster in the range of $[N_l, N_h]$.
2. Label cluster E_i with center c_i as positive if $c_i^x < \theta - r_i$ and negative if $c_i^x > \theta + r_i$, where c_i^x is the X-axis value of the center c_i and θ is the threshold value between c_l and c_h. Remove the clusters which are not assigned to either of the classes.
3. Generate the data points for each cluster according to an n-dimensional normal distribution whose mean is the center c and covariance matrix is $r^2 I$. The class label of each point is the label of the parent cluster.

A two-dimensional dataset was generated with parameter values $k = 50$, $c_l = 0.0$, $c_h = 1.0$, $r_l = 0.0$, $r_h = 0.1$, $N_l = 0$, $N_h = 10,000$, and $\theta = 0.5$. The results are tabulated in Table 6. The first column contains results using CB-SVM as reported in [6]. The results show that our algorithm is on par with CB-SVM. The test accuracy remains the same (99%) with approximately the same training time. Also,

Table 7
Results for *k*-NNC.

Dataset	Synthetic dataset 1	Synthetic dataset 2	OCR
Time taken by *k*-NNC	**1.695**	**17.347**	0.224
Accuracy (*k*-NNC)	**100**	**97.1**	92.49
NumClusters	**6**	**6**	15
Threshold	**5**	**5**	5
Time taken by *k*-NNC with 2-level *k*-means	**1.382**	**9.852**	0.2
Accuracy(*k*-NNC with 2-level *k*-means)	**100**	**96.8**	92.40

when a slight reduction in the test accuracy is allowed through the adjustment of the parameters NumClusters(k) and Threshold(τ), the training time reduces significantly (4.62 s with an accuracy of 96% and 2.91 s with an accuracy of 95%).

6.3. Two-level k-means in k-NNC

The results corresponding to the integration of two-level *k*-means in *k*-NNC are presented in Table 7. We selected 1000 test patterns from each of the synthetic datasets. Since *k*-NNC is a multi-class classifier, the entire OCR dataset was used. For each dataset, we build a 5-NNC and record the average time taken (in seconds) for the classification of a single example and the test set accuracy. A considerable reduction in the time is observed, especially for synthetic dataset 2 (classification time reduces from 17.347 to 9.852 s), which proves that our algorithm shows better performance on large datasets.

7. Conclusion and future work

The proposed two-level *k*-means algorithm would be very useful for handling large datasets. A significant speedup is achieved in the clustering process. Analytical and empirical results also show that a good reduction in the computational complexity can be achieved while maintaining the same generalization performance by incorporating this algorithm in the training process of classifiers. There are many domains in which there is a requirement for algorithms which can handle large datasets. Some of the domains in which the proposed algorithm would prove to be useful are:

1. *Topic-based web search*: Existing classifiers cannot handle large repositories such as the World Wide Web. Finding documents which are based on or are related to a particular topic involves categorizing and sifting through large index files and requires fast classifiers.
2. *Spam filtering*: Spam filtering translates to a two-class classification problem with a large training set, the two classes being *Spam* and *Not Spam*. An effective spam filtering system with low running time and good accuracy can be built using our technique.
3. *Communication network analysis*: Packet analysis and intrusion detection in communication networks also requires fast and efficient classification mechanisms.

We have observed experimentally that the two-level *k*-means algorithm gives the best performance when the input dataset is uniformly distributed. This is due to the fact the *k*-means algorithm is less likely to generate non-uniformly sized clusters when the data is uniform [15]. We are currently working on reducing the effect of the input data distribution on the performance of the algorithm. We are also studying the effect of integrating the two-level *k*-means algorithm into other discriminative classifiers such as Decision Trees. In this paper, we have established the theoretical basis for the reduction in the complexity and the training time. The relationship between the generalization performance of the classifier before and after the incorporation of the clustering algorithm is to be established.

References

[1] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Computing Surveys 31 (3) (1999) 264–323.
[2] S. Lloyd, Least squares quantization in PCM, IEEE Transactions on Information Theory 28 (2) (1982) 129–137.
[3] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, A.Y. Wu, An efficient k-means clustering algorithm: analysis and implementation, IEEE Transactions on Pattern Analysis and Machine Intelligence (2002) 881–892.
[4] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: The Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, 1967, pp. 281–297.
[5] S. Guha, A. Meyerson, N. Mishra, R. Motwani, L. O'Callaghan, Clustering data streams: theory and practice, IEEE Transactions on Knowledge and Data Engineering (2003) 515–528.
[6] H. Yu, J. Yang, J. Han, Classifying large data sets using SVMs with hierarchical clusters, in: The Proceedings of the Ninth ACM SIGKDD international conference on Knowledge discovery and Data Mining, 2003, pp. 306–315.
[7] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: an efficient data clustering method for very large databases, in: The Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, 1996, pp. 103–114.
[8] B. Zhang, S.N. Srihari, Fast k-Nearest neighbor classification using cluster-based trees, IEEE Transactions on Pattern Analysis and Machine Intelligence (2004) 525–528.
[9] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[10] C.J.C. Burges, A tutorial on Support Vector Machines for pattern recognition, Data Mining and Knowledge Discovery, vol. 2, Springer, Berlin, 1998, pp. 121–167.
[11] T. Joachims, Making large-scale SVM Learning practical, in: Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, USA, 1999, pp. 169–184.
[12] J. Platt, Sequential minimal optimization a fast algorithm for training support vector machines, in: Advances in Kernel Methods: Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 185–208.
[13] I.W. Tsang, J.T. Kwok, P.M. Cheung, Core vector machines: fast SVM training on very large data sets, The Journal of Machine Learning Research 6 (2005) 363–392.
[14] T. Joachims, Training linear SVMs in linear time, in: The Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006, pp. 217–226.
[15] H. Xiong, J. Wu, J. Chen, K-means clustering versus validation measures: a data distribution perspective, IEEE Transactions on Systems, Man, and Cybernetics. Part B Cybernetics: A Publication of the IEEE Systems, Man, and Cybernetics Society 39 (2) (2009) 318–331.

About the Author—RADHA CHITTA received her BE degree in Computer Science and Engineering from PES Institute of Technology, Bangalore, India and ME degree in System Science and Engineering from the Indian Institute of Science, Bangalore, India. She is currently working in Yahoo! Research and Development, Bangalore, India. Her research interests include data mining and pattern recognition.

About the Author—M. NARASIMHA MURTY received his BE, ME and PhD degrees from the Indian Institute of Science, Bangalore, India. He is currently a professor in the Department of Computer Science and Automation. His research interests are in pattern clustering and data mining.