Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 4082 – 4101

doi:10.3906/elk-1807-163

A depth-based nearest neighbor algorithm for high-dimensional data classification

Sandhya HARIKUMAR*, **Akhil ARAVINDAKSHAN SAVITHRI**, **Ramachandra KAIMAL**
Department of Computer Science and Engineering, Amrita Vishwa Vidyapeetham, Amritapuri, India

Received: 17.07.2018 • **Accepted/Published Online:** 05.05.2019 • **Final Version:** 26.11.2019

Abstract: Nearest neighbor algorithms like k-nearest neighbors (kNN) are fundamental supervised learning techniques to classify a query instance based on class labels of its neighbors. However, quite often, huge volumes of datasets are not fully labeled and the unknown probability distribution of the instances may be uneven. Moreover, kNN suffers from challenges like curse of dimensionality, setting the optimal number of neighbors, and scalability for high-dimensional data. To overcome these challenges, we propose an improvised approach of classification via depth representation of subspace clusters formed from high-dimensional data. We offer a consistent and principled approach to dynamically choose the nearest neighbors for classification of a query point by i) identifying structures and distributions of data; ii) extracting relevant features, and iii) deriving an optimum value of k depending on the structure of data by representing data using data depth function. We propose an improvised classification algorithm using a depth-based representation of clusters, to improve performance in terms of execution time and accuracy. Experimentation on real-world datasets reveals that proposed approach is at least two orders of magnitude faster for high-dimensional dataset and is at least as accurate as traditional kNN.

Key words: Subspace-clustering, data-depth, information gain, nearest neighbor, classification

1. Introduction

The k-nearest neighbors (kNN) algorithm is a simple nonparametric classification technique which is efficient, provided it is given a good distance metric and has enough labeled training data [1]. If we have no knowledge of the underlying distribution, a decision to classify data point x into a class y depends on a set of known samples $(x_1, y_1), ..., (x_n, y_n)$. Recommendation systems, text categorization, heart disease classification, and market predictions are some of the applications of classification algorithms [2, 3]. However, with the advent of high-dimensional data in rapidly developing domains such as medicine, finance, education, and genomics, the unknown data distributions may be uneven which may affect the accuracy of classification results. Furthermore, available data points may not be labelled fully. The existing techniques for classification of such datasets are either not scalable or not accurate due to failure in identifying intricate relationships existing amongst the features of data. Thus, traditional algorithms suitable for low-dimensional data, need to be improvised to suit the existing scenarios.

One advantage of kNN over other algorithms is that it can be applied to data following any distribution [4–6]. However, since it does not understand the data, its successful execution requires careful choice of parameters k, feature weights w, and distance measure [7]. Most of the works focus on setting of k and w irrespective of structure and distribution of data whose labels are to be found. Furthermore, kNN suffers from 'curse of

*Correspondence: sandhyaharikumar@am.amrite.edu

dimensionality' and sparsity in higher dimensions that lead to expensive computational cost and high storage requirements for voluminous data [4, 5, 8]. Hence, an efficient algorithm for classification of high-dimensional data is required.

For instance, consider Figure 1 that shows two different distributions of instances in $2d$ space. A query point q is to be classified to one of the two distributions, but since q is equidistant from the centers of both the distributions, any value for k may lead to wrong classification. Similar is the case with high-dimensional data where all points seem equidistant [8]. Thus, a good clustering mechanism is required to properly model the data distributions with relevant features before applying kNN. Quite often, though the data is high-dimensional, there exists many low-dimensional structures corresponding to several classes or categories. The points that belong to a particular class are usually more similar to each other than points that belong to other classes in these low-dimensional structures. Traditional clustering algorithms find clusters in the entire space of the dataset. However, if the data is high-dimensional, such clusters may be difficult to comprehend. Hence, there is a need for a different approach of clustering that finds clusters in the subspaces of the high-dimensional dataset, called subspace clustering [9–12]. Various types of subspace clustering methods exist in the literature such as CLIQUE [13], SUBCLU [14], and MAFIA [15] that finds clusters in all subspaces. There are other types of subspace clustering methods called projective clustering such as PROCLUS [16, 17], FIND-IT [18], IGSC [19] and δ-CLUSTERS [20] that find clusters in axis-parallel subspaces. We follow a previous work where information gain-based semi-supervised-subspace clustering (IGSC) is suggested [19] to model the distributions of high-dimensional data with relevant features. The details of IGSC algorithm are given in Section 4 and the algorithm is given in Algorithm 2. The different subspace clusters so obtained may have different features. Hence, a unique representation of all the clustered data is required to undermine the effect of subspaces. This is achieved by a concept of data depth function [21] explained in Section 3.2

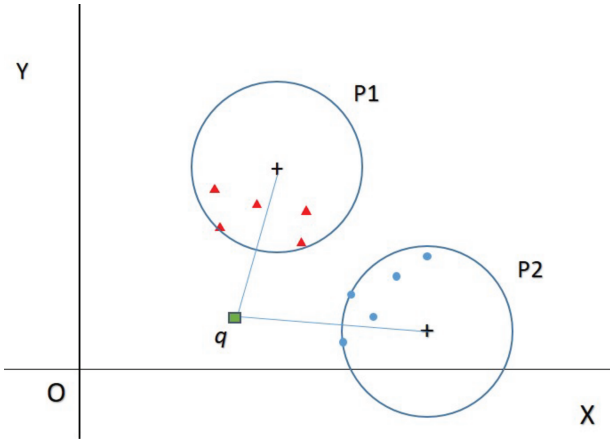


Figure 1. The query point q equidistant from the centers of two distributions P1 and P2.

Looking for k-nearest neighbors for a new query example from a set of n data points has a straightforward algorithm with cost $O(nlog(k))$ [1]. This is biased towards the choice of k and is quite expensive if the number of data points, n, and the number of features per data point, d, is too large. Another approach is to use indexing with k-d trees [22], which fails for high-dimensional data since it requires 2^d examples for efficient index. Furthermore, the kNN algorithm is inefficient without multivariate analysis and premodelling, where dynamic classification is needed for large repository [23].

In this paper, we offer a consistent and principled approach to dynamically choose the nearest neighbors for classification of a query point by *i)* identifying structures and distributions of data, *ii)* extracting relevant features, *iii)* deriving an optimum value of k depending on the structure of data by representing data using data depth function. Thus, given a query point, q, we aim to find the best distribution of points, S, with relevant features, that matches q, so as to minimize the distance between q and S to effectively retrieve appropriate label for q. For this, we adapt an existing subspace clustering called IGSC to obtain informative clusters, each with relevant dimensions using information gain [19]. Thereafter, we propose an improvised classification algorithm using a depth-based representation of subspace clusters, to improve scalability and accuracy of classification for high-dimensional data. Experimental results presented in Section 5 reveals that our approach is at least two orders of magnitude faster for high-dimensional dataset and is at least as accurate as conventional kNN. Thus, our contributions are as follows:

1. Identify different low-dimensional structures and varied distributions of partially labelled data consistent with the available labels, by adapting subspace clustering technique, IGSC [19].

2. Apply the concept of data depth function on each of the low-dimensional structures identified, to give each of them a unique representation by eliminating the influence of subspaces in individual clusters.

3. Propose an augmented nearest neighbor algorithm based on L_1 data depth function for classification of high-dimensional data on this newly formed depth space.

The rest of the paper is organized as follows. Section 2 focuses on related work on kNN and its variants. Section 3 gives background and few important definitions and concepts. Section 4 presents the problem formulation and the devised algorithms. Section 5 shows experimental results and performance evaluation. Finally, Section 6 concludes the paper.

2. Related work

In practical scenarios, nonparametric techniques for classification, based on nearest neighbor strategy, such as kNN [6] and parzen window classifier [24], are best suited for data whose distributions are either unknown or that have different types of distributions [4]. Parametric techniques for high-dimensional classification such as support vector machine (SVM) depends on the selection of the kernel function parameters, which is a major drawback. Feature selection models that are computationally tractable is proposed in [25, 26] for classification of high-dimensional data using SVM. However, the approach is only suitable for data that is linearly separable. Hastie and Tibshirani [27] developed an adaptive method of nearest neighbor classification by using local discrimination information to estimate a subspace for global dimension reduction. Parzen window classifier was deployed in [28] to devise an active learning algorithm in order to address the problem in which large amounts of unlabeled data are available. Parzen window classifier depends on a single parameter: the kernel window size, which can be determined by cross-validation. In all these nonparametric approaches, the parameter k for kNN and window size for Parzen window, is critical. Thus, if a strategy to explore the interrelationships amongst data objects in subsets of features is applied, the crucial choice of k and window size can be made dynamic depending on the varying distributions of data.

This leads to the necessity of devising an efficient strategy for neighborhood algorithm by exploring intricate relationships amongst the data objects in subsets of features. Such a problem requires clustering approaches that explore various data objects to find clusters in different subsets of features. One such strategy is subspace clustering [9–11]. However, the subspace clusters may have different features in different clusters.

Hence, a unique representation of all the clustered data is required to undermine the effect of subspaces. We brief the trends in subspace clustering and data-depth-based classification by reviewing some of the relevant works in this area. We try to bring out the essence of existing algorithms and explain how our approach differs from the state-of-the-art approaches.

Subspace clustering seeks clusters in different subspaces within a dataset. Based on the approach of finding subspaces, subspace clustering (SC) can be classified into soft subspace clustering (SSC) and hard subspace clustering (HSC). While HSC like CLIQUE [13], SUBCLU [14], and MAFIA [15] follow a bottom-up approach to find exact subspaces, PROCLUS [16, 17, 29], FIND-IT [18], and δ-CLUSTERS [20] follow a top-down approach by assigning weights to dimensions in each cluster. SSC can be considered an extension of feature weighting clustering [19, 30–34]. The features in each subspace cluster may be different. Hence, for classification, a unique representation of data so as to reduce the effect of subspaces is required for effective classification. This can be achieved by data depth function [21] that orders data in a central-outward order where each point is then identified by its depth in the representation.

Data depth representation was first proposed in [21], where Vardi and Zang proposed the idea of L1-median and L1 data depth. The depth of a point relative to a given dataset measures how deep that point lies in the data cloud. The data depth concept provides center-outward ordering of points in any dimension, and leads to a new nonparametric multivariate statistical analysis, in which no distributional assumptions are needed [35]. The paper gives three related results. The first one is, a new simple, fast, and monotonically converging algorithm for deriving the L1-median of a distribution in R^d. The second is, a new general definition for depth functions in terms of multivariate medians. The third is, a simple closed form formula of the L1-depth function for a given data cloud in R^d. A depth-based modification of k-nearest neighbor method was proposed by Ondrej Vencalek in [36]. The newly proposed method is based on the idea that points with similar location with respect to two considered distributions will have similar depths with respect to the distributions. Depth-based classification is also proposed in [37, 38] to avoid the parametric assumptions and to get better classifier for skewed data. Depth-based classifiers are affine invariant; hence, the classification methods do not change, for example, with the change in units (scale) of data [39]. Depth functions might be very slow in high-dimensional data. Thus, our classification approach models the data using a subspace clustering technique, that reduces the dimensions in each cluster and then depth function is applied. Reliable depth-based classifiers are available in R-package ddalpha [40] but not for very high-dimensional data. For high-dimensional data up to 100, an approach based on transformation of the original data to the DD-space using spatial or localized spatial depth is proposed in [41]. After transformation, these depths are used as explanatory variables in multinomial additive logistic regression model.

In all these works, data depth of objects in subspace clusters is not leveraged. Moreover, the use of labels, if available, are not taken into consideration for the formation of clusters. If data labels are available then leveraging such information for subspace clustering can not only be useful for validating clusters but can also enhance the learning process to be consistent with the available knowledge. Hence, our work adopts entropy and information gain with respect to the knowledge about the available labels to guide the significance of important attributes leading to subspace clusters.

3. Improvised classification algorithm

Our proposed classification approach is based on a principled combination of subspace clustering [7, 9, 11, 19, 28], data depth representation [21, 35, 36], and nearest neighborhood using an optimization function. Here we

specifically divide classification of high-dimensional data into three phases: (i) subspace clustering for feature-based cluster formation, (ii) formation of data depth space for unique representation of data irrespective of subspaces, and (iii) nonparametric classification of a given query point. The block diagram of the proposed work is given in Figure 2.

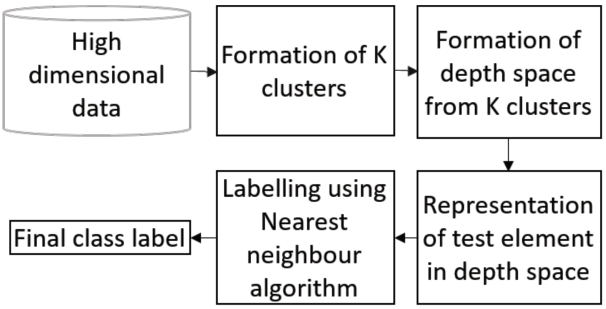


Figure 2. Block diagram of the proposed nearest neighbor algorithm.

3.1. Phase 1: subspace clustering based on entropy and information gain

Let D be a dataset in $\mathbb{R}^d$ with each instance $t \in D$ defined by a set of attributes $A = \{a_1, ..., a_d\}$. Each instance t is labeled with $y = p_i$ or unlabeled with $y = 0$, where $p_i \in P = \{p_1, .., p_c\}$, with c being the total number of class labels. We define subspace cluster S_i as a cluster of instances that are most similar to each other along $A_i \subset A$ attributes. $|A_i| = l$ and $l \leq d$. Each cluster has a representative element $m_i \in D$, called medoid, with label y. In this context, an objective function F is devised to find an optimal set of K subspace clusters $S = \{S_1, ..., S_K\}$ so as to reduce the entropy of each subspace cluster S_i in A_i dimensions with m_i as the medoid. The subspaces so formed are axis aligned subspaces [16].

Formally, let K denote the number of clusters, $M \subset D$ denote the set of potential representative elements called medoids, and P denote the set of class labels with $|P| = c$. Each subspace cluster S_i is defined by a subset of dimensions $A_i \subset A$ and a medoid $m_i \in M$. Then, the objective function F is defined as a minimization function given by (1)

$$F(K, M, P) = \text{minimize} \sum_{i=1}^{K} H(S_i, P) \frac{|S_i|}{|D|}, \tag{1}$$

where

$H(S_i, P) = -\sum_{p_j \in P} Pr(p_j, S_i) log_2 Pr(p_j, S_i)$

$Pr(p_j, S_i) = \frac{1}{|S_i|} \Sigma_{k=1}^{N} \mathbf{1_{p_j}}(\mathbf{x_{ki}})$

Here $H(S_i, P)$ is the entropy of the cluster S_i with dimensions A_i and medoid m_i with respect to the available set of labels P. $Pr(p_j, S_i)$ is the probability of an instance $x_{ki} \in S_i$ being classified as p_j. The details of finding medoids m_i and composing A_i for each subspace cluster S_i are given in Section 4.

3.2. Phase 2: formation of data depth space

The concept of data depth was introduced in [42]. As the name suggests, data depth represents the depth of a data point with respect to the given probability distribution of each subspace cluster. The depth of a point relative to a given dataset measures how deep that point lies in the data cloud. The data depth concept provides

center-outward ordering of points in any dimension and leads to a new nonparametric multivariate statistical analysis in which no distributional assumptions are needed [21, 35, 36]. It is a very useful tool for nonparametric multivariate data inference. Data depth provides some ordering of points in the multidimensional space with respect to some probability measure defined on the space. Those functions that generate such ordering are known as depth functions. According to [35, 42], there are four main properties of a depth function as follows.

- Affine invariance: According to this property, the depth of a certain point x should not change while performing any transformation. Thus, it should not depend on the underlying coordinate system.
- Vanishing at infinity: This property suggests that the depth of a point x should be very small when the point is far away from others.
- Maximality at center: Consider a symmetric probability distribution with a unique point which can be called the center of symmetry, then that point should be the deepest point, i.e. the depth function should attain its maximum value there.
- Monotonicity relative to deepest point: As a point $x \in R^d$ moves away from the 'deepest point' (the point at which the depth function attains maximum value; in particular, for a symmetric distribution, the center) along any fixed ray through the center, the depth at x should decrease monotonically.

Vardi and Zang proposed the idea of $L_1 - median$ and $L_1 - depth$ [21]. $L_1 - depth$ is defined as follows.

Definition : $L_1 - median:$ Let P be a probability distribution of a random vector X on $\mathbb{R}^d$ such that $E_p||X|| < \infty$, where $||.||$ denotes the Euclidean norm in $\mathbb{R}^d$. The $L_1 - median$ of the distribution P is defined as in Eq. (2):

$$M(Q) = \arg min_{y \in \mathbb{R}^d} E_P||X - y||. \tag{2}$$

The $L_1 - depth$ of a point $x \in \mathbb{R}^d$ with respect to the distribution P is defined as in Eq. (3):

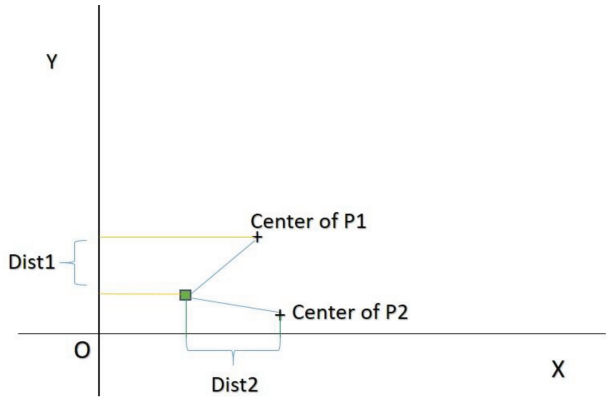
$$D(x;P) = 1 - \inf\{w >= 0 : M(\frac{w}{1+w}\delta_x + \frac{1}{1+w}P) = x\}, \tag{3}$$

where δ_x is a point mass at x. That is, $1-D(x;P)$ is the minimum incremental mass w needed at x for x to become the L_1–median of the mixture $(\frac{w}{1+w}\delta_x + \frac{1}{1+w}P)$.

In our proposed model, we use the concept of $L_1 - depth$ so that the second phase shifts informative subspace clusters generated in the first phase to a depth space using L1-depth function[21]. In order to do an impartial classification, we need to get rid of the influence of subspaces in individual clusters. Hence, the clusters need to be represented in such a way that, instead of their original features, each element now gets shifted to a feature space where each feature represents the "amount of influence" from individual clusters in the clustering as shown in Figure 3. This "amount of influence" is known as "depth" of data point. Even the test sample will be shifted to this depth space so that we will be able to do an impartial classification. Finally, we adapt kNN in this newly generated depth space as shown in Figure 4.

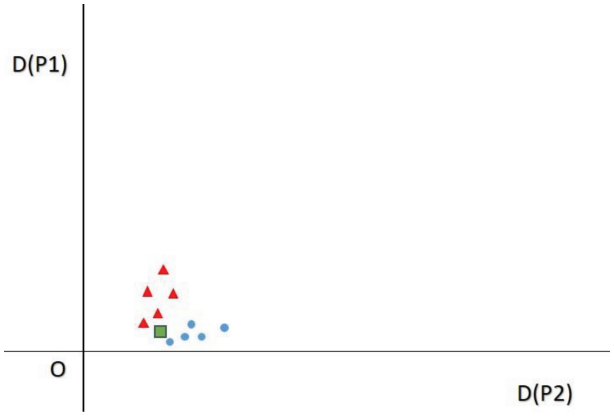
Let S = $\{S_1, S_2, ..., S_K\}$ be the set of final subspace clusters, from phase one, with important attributes $A_i \subset A$ where $1 \leq i \leq K$. Each data point s is represented by K clusters where $s^{|A_i|} \rightarrow s^K$ is as given in Eq. (4):

$$D(s) = [D(s, S_1, A_1), \cdots, D(s, S_K, A_K)] \tag{4}$$



Depth of data point P with respect to P1 and with respect to P2 using L1-depth function

Figure 3. Depth of data point q with respect to center of cluster $P1$ and with respect to center of cluster $P2$ using L1-depth function.



Depth representation of all points with respect to 2 clusters

Figure 4. Depth representation of all points with respect to 2 clusters.

$D(s, S_i, A_i)$ is the L1-depth of $\boldsymbol{s}$, with respect to each cluster S_i and respective dimension set A_i as in Eq. (5).

$$D(s, S_i, A_i) = 1 - E_P \left\| \frac{s^{|A_i|} - S_i^{|A_i|}}{\left\| s^{|A_i|} - S_i^{|A_i|} \right\|} \right\|, \tag{5}$$

where S_i is the set of instances in i^{th} subspace cluster. $s^{|A_i|}$ restrict the dimensions of s from A to A_i with $|A_i| = l_i$. E_P is the expectation with respect to the distribution S_i.

Figure 4 shows the depth representation of all instances in Figure 1 with respect to 2 clusters.

3.3. Phase 3: classification of points in depth space

The third phase classifies a test data instance by applying nearest neighbor algorithm on the depth space, formed in second phase. Each test data instance is also represented in the depth space by following (4). This test data instance is initially compared with the cluster medoids to find the nearest medoid. Once the nearest medoid is found, nearest neighbor algorithm is applied in the respective subspace cluster to find the class label of the test instance based on majority voting.

4. Algorithms

As explained in previous subsection, the proposed approach consists of three phases. The first phase identifies subspace clusters in the dataset, the second phase performs the depth space representation of generated clusters, and the third phase performs classification by applying nearest neighbor algorithm on depth space. The notations used in the algorithms are described in Table 1.

Algorithm 1 gives an outline of the proposed strategy.

For subspace clustering, we adapt an iterative approach of IGSC [19]. The outline of IGSC is shown in Algorithm 2.

The main difference in this algorithm as compared to other subspace clustering algorithms is the objective function and the strategy to find subspace for each cluster based on entropy and information gain. Furthermore, our subspace clustering involves finding a potential set of medoids that can form the representatives for subspace

Table 1. Notations used in the algorithm.

Notations	Description
D	Partially labeled numerical dataset
A	Set of attributes in D. $A = \{a_1, ..., a_d\}$
K	Number of subspace clusters
k	Number of nearest neighbors
l	Avg. number of dimensions per subspace cluster
T	Test dataset
C	Set of class labels for each data point in T
S	Set of subspace clusters. $\{S_1, ..., S_K\}$
A_i	Set of dimensions associated with a subspace cluster S_i
$\tilde{A}$	Set of A_i's for each subspace cluster S_i.$\{A_1, ..., A_K\}$
M_{curr}	Medoid set from the current iteration
M_{best}	Best set of medoids found so far
δ	Distance between two instances.
E,F	Constant integers
V	Points at δ distance from each medoids

Algorithm 1: Depth-based classification

Input : $D, K, l, T = \{T_1, \cdots, T_M\}$
Output: $C = \{C_1, \cdots, C_M\}$

1 $S, \tilde{A} \leftarrow IGSC(D, K, l)$ `// Information gain based susbpace clustering`

2 $D_p \leftarrow Depth(S, \tilde{A})$ `// Depth of each data point in subspace clusters as per Eqs.` 4 `and` 5

3 $C \leftarrow L_1 depthNN(D_p, S, \tilde{A}, T)$ `// Classification in depth space`

cluster. If K clusters are required to be formed from dataset D, we choose more than K representatives, called medoids. As given in Algorithm 2, $e * K$ medoids are chosen, where e is a positive integer constant. We choose more medoids than the required number of clusters, to ensure that during iterative process of finding subspace clsuters, if some outliers are formed, those can be discarded and better subspace clusters can be found. A greedy technique has been used to generate optimal set of medoids [16]. The greedy algorithm adapted from [16] is given in Algorithm 3. We reproduce the algorithm here for ease of understanding the search for potential medoids.

One of the important aspects of this clustering is to identify important subset of features for each subspace cluster. We use the concept of information gain and entropy [43] for subspace search as given in [19]. Our approach makes use of labels to identify important attributes in each cluster. This helps in forming subspace clusters consistent with the labeled dataset, and aid in modifying and extending the labels to reflect the hidden structures in low-dimensional data. The proposed model searches for a good partition by formulating the objective function that aims at minimizing the entropy of each cluster.

The relevant features are found by computing information gain of each feature that helps in deciding the label of an instance. Let S_i be a cluster represented by medoid m_i. Let $A = a_1, ..., a_d, y$ be the set of features

Algorithm 2: Information-gain-based subspace clustering: $IGSC(D, K, l)$

Input : D, K, l
Output: $S = \{S_1, \cdots, S_K\}, \tilde{A} = \{A_1, \cdots, A_K\}, M_{best} = \{M_1, \cdots, M_K\}$

$M \leftarrow$ Medoids$(D, e * K)$;// Find $e * K$ potential medoids from D where e is a positive integer constant
$BestObjVal \leftarrow \infty$;
$M_{curr} \leftarrow$Get K medoids from M randomly;
while *(BestObjVal not converged)* **do**
  for $m_i \in M_{curr}$ **do**
    $\delta_i \leftarrow \arg\min_{m_j \in M_{curr}, i!=j} |m_i - m_j|$;
    $V_i \leftarrow$ Points at distance δ_i from m_i ;
  end
  $V \leftarrow \{V_1 \cdots V_K\}$;
  $dimensionSets \leftarrow$ ExtractDim(K, l, V)// as per $Eqn.$ 6, 7, 8
  $S \leftarrow$ Assign$(dimensionSets)$;
  $Obj \leftarrow$ ClusterEval$(S, dimensionSets)$;
  if $Obj < BestObjVal$ **then**
    $BestObjVal \leftarrow Obj$;
    $M_{best} \leftarrow M_{curr}$;
  end
  Replace bad medoids in M_{best} ;
  $M_{curr} \leftarrow M_{best}$;
end
$\tilde{A} \leftarrow dimensionSets$;
Refine$(S, \tilde{A}, M_{best})$;

where y is the label feature. Each instance $x_k \in S_i$ has $y_k \in P = \{p_1, ..., p_c\}$. In order to find relevant features with respect to the class label of the instances in S_i, we find the entropy of S_i and information gain [43] of each feature a_i. This method is similar to the one adopted in [19]. Formally, entropy and information gain are defined as follows.

Entropy : Let D be a set of N examples and let P be a set of c classes. Let $Pr(p_i, D)$ be a fraction of instances in D, which belongs to class $p_i \in P$. Then, the entropy of D is computed as given in Eq. (6). Higher entropy value of D indicates higher information content in D.

$$Entropy(D) = -\sum_{i=1}^{c} Pr(p_i, D) \times log(Pr(p_i, D)). \tag{6}$$

Let D_i be the set of examples having value v_i for attribute a_i. Let v be the number of distinct values

Algorithm 3: Finding potential medoids: $Medoids(D, z)$

Input : Dataset : $D = \{D_1, ..., D_n\}$, Required number of medoids z
Output: Potential medoids $M = \{m_1, ..., m_z\}$
Initialize $M = \{m_1\}$, where m_1 is a random point from D;
Initialize $Dist = \{d_1, ..., d_n\}$ where d_i is distance between $t_i \in D \setminus M$ and m_1;
for $i = 2$ *to* z **do**
    Choose a point $t_j \in D \setminus M$ having highest d_j in $Dist$;
    `// Farthest point from existing medoid`
    $m_i \leftarrow t_j$;
    $M = M \cup \{m_i\}$;
    Compute distance of each $t_i \in D \setminus M$ to the closest medoid in M;
    Update $Dist$ to latest distance for each point t_j;
end
return M;

of attribute a_i. Then, the average entropy of a_i is given as in Eq. (7)

$$AvgEntropy_{(a_i)}(D) = -\sum_{i=1}^{v} \frac{|D_i|}{|D|} \times Inf(D_i). \tag{7}$$

Our goal is to find which attribute in D is most useful for finding the class y of an example $x_k \in D$. We use information gain [43] to find significance of each attribute $a_i \in A$ as given in Eq. (8).

$$Information\_Gain(a_i) = Entropy(D) - AvgEntropy_{(a_i)}(D). \tag{8}$$

Thus, significance of each feature a_i is found. The feature with the highest information gain is the most relevant one. Thus, instead of finding all combinations of l features, we chose l relevant features in the decreasing order of their information gain, thereby reducing the complexity of the search for important features to form a subspace cluster. Algorithm 4 presents the search algorithm for finding l relevant attributes for each subspace $S_i \in S$.

Once the subspace clusters are formed, the entire clustering is shifted to a depth space where each instance is described in terms of depth with respect to individual clusters. This helps in alleviating the influence of different subspaces in each cluster. We use here L1 depth since this is robust for high-dimensional data [44]. In the depth space, nearest neighbor algorithm is applied to find the label of a given test instance. Algorithm 5 shows the classification in L_1 depth space.

5. Experimentation and analysis

5.1. Experimental design and setup

The experimentation was conducted in two phases with information-gain-based subspace clustering in phase one and L1 depth-based kNN classification in the second phase. We used a commodity machine with 2.8 GHz/1.9 GHz AMD Quad-Core processor and 6 GB main memory. Weka 3.8 together with OpenSubspace (Weka Subspace-Clustering Integration) was used for comparing various algorithms with our approach. The performance was measured in terms of scalability and classification accuracy.

Algorithm 4: Extracting relevant dimensions: $ExtractDim(V, A, l)$

Input : $V = \{V_1, ..., V_K\}, A = \{a_1, ..., a_d\}, l$
Output: Relevant dimensions of each partition $V_j \in V$

Let $Info\_Arr[1...K][1...d]$ be a matrix to store information gain of attributes for each partition V_j;
for $V_j \in V$ **do**
  $t \leftarrow 1$;
  $EV_j \leftarrow$ *Entropy of* V_j *as per Eq.* (6);
  for $a_i \in A$ **do**
    $Avg\_a_i \leftarrow$ *Average Entropy of attribute* a_i *as per Eq.* (8) $IG_{ai} \leftarrow EV_j - Avg\_a_i$;
      `// Information gain of each attribute`
    $Info\_Arr[j][t++] \leftarrow IG_{ai}$
  end
  Sort $Info\_Arr[j]$ `// Sort information gain of attributes in` V_j `in descending order`
end
return $Info_Arr$;

Algorithm 5: Classification in L_1-depth space: $L_1 depthNN(D_p, S, \tilde{A}, T)$

Input : $D_p, S, \tilde{A}, T$
Output: Class labels of individual instances in T

for $t \in T$ **do**
  $depth_t \leftarrow Depth(t, S, \tilde{A})$ `// as per Eq. (5)`
end
for $t \in T$ **do**
  $S_j \leftarrow$ *Subspace cluster nearest to* $depth_t$;
  $D_j \leftarrow$ *Depth space of* S_j *points from* D_p;
  $label_t \leftarrow$ kNN$(depth_t, D_j, |S_j|)$ `// Find` k `nearest neighbors of test instance` $depth_t$
    `in depth space` D_j `where` k `can take a maximum value of` $|S_j|$
end

5.2. Phase 1: experimental analysis of proposed subspace clustering approach

In this section, we present our experimental results and comparison of our subspace clustering algorithm with a subspace clustering algorithm called PROCLUS as well as with a traditional clustering approach K-means, to show the effectiveness of our proposed work. To evaluate the effectiveness and performance of our proposed method, 13 publicly available datasets from UCI repository (http://archive.ics.uci.edu/ml/index.php) were used. These 13 real world datasets are high-dimensional with dimensionality varying from 9 to 27,680 and number of instances from 34 to 8000 as shown in Table 2. For scalability experiments, we replicated the instances of Pendigits and Synthetic datasets up to $400,000$ records. The size of each dataset is increased by an iterative process. If the size of dataset is X and the required size is Y, then $\lceil (Y/X) \rceil$ iterations are required. During each iteration, each instance is inserted into the dataset only once to avoid skewed data. However, each duplicate instance will be treated as a separate instance in the dataset by giving a unique ID. Hence, we ensure that our algorithm considers Y records separately and does the clustering irrespective of duplicate values in the instances.

We used the following performance parameters in our empirical study: size of the dataset $|D|$, number of subspace clusters K, and average dimensionality of each subspace cluster l to demonstrate the effectiveness of

Table 2. Datasets obtained from UCI repository used for experimentation.

Dataset	Instances	Attributes	Classes
B-Cell1	45	4027	2
B-Cell2	96	4027	11
B-Cell3	96	4027	9
Colon	62	2001	2
Embryonal tumer	60	7130	2
Lukemia1	34	7130	2
Lukemia2	38	7130	2
Pendigits	8000	17	10
Synthetic	1596	76	13
ECML	90	27,680	43
GCM	144	16,064	14
Diabetes	768	9	2
Glass	214	10	6
German	1000	24	2
Bank	41,188	62	2

our proposed algorithm. For scalability, we replicated few datasets like Pendigits and Synthetic datasets both in terms of dimensions and number of instances. The proposed approach was then applied on the datasets to determine how the running time and the cluster quality varied with $|D|$, K, and l.

The experiments were repeated 15 times each and average values were considered for final evaluation. The proposed algorithm is compared with other algorithms on different datasets. Metrics used for validation purpose are purity, accuracy based on F1 measure, entropy, and Davies–Bouldin index (DBI) measures.

5.2.1. Results and analysis

Here we discuss cluster quality. DBI [45] is an internal validation metric used for evaluating clustering algorithms irrespective of the target of the instances in a dataset. It just uses the available inherent features of the dataset and the clusters found. For a given clustering, we customized DBI as given in Eq. (9):

$$DBI(K,M) = \frac{1}{K}\sum_{i=1}^{|K|} \underset{j\neq i}{max} \frac{SC(S_i)+SC(S_j)}{SP(S)_{i,j}}, \tag{9}$$

where $SC(S_i) = \frac{1}{|S_i|}\sum_{j=1}^{|S_i|}\left|t_j^{A_i} - m_i^{A_i}\right|$ and $SP(S)_{i,j} = \left\|m_i^{A_i\cup A_j} - m_j^{A_i\cup A_j}\right\| = \sum_{u\in(A_i\cup A_j)}|a_{u,i} - a_{u,j}|$

Here K is the total number of clusters, $t_j^{A_i}$ is A_i dimensional instance assigned to S_i and m_i is the medoid of cluster S_i. $a_{u,i}$ denotes value of u^{th} attribute of m_i. DBI always has a value > 0. The lower the DBI value is, the higher the clustering quality is as is evident from Eq. (9):

As for purity, it (nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html) is an external validation metric that is based on the targets of the instances. This metric checks how well the clustering matches the available targets/classes. For a clustering with S=$\{S_1,\cdots,S_K\}$ clusters and P=$\{p_1\ ,\cdots,p_c\}$

labels, purity can be calculated as in Eq. (10).

$$purity(S,P) = \frac{1}{N}\sum_{i} \underset{j}{max} \left|S_i \cap p_j\right|. \tag{10}$$

The range of purity values is in the interval $[0,1]$. The higher the purity value is, the higher the clusters match with the available targets/classes. Thus, we can infer that the clustering quality is high for higher values of purity. Table 3 shows the purity and DBI measures of subspace clusters formed.

Table 3. Comparison of purity and DBI measures of generated clusters using proposed approach with K-means clustering in full dimensions and a subspace clustering PROCLUS.

	Purity			DBI		
Dataset	PROCLUS	Proposed approach	K-means	PROCLUS	Proposed approach	K-means
B-Cell1	0.6	0.84	0.6	1.60	1.67	2.80
B-Cell2	0.43	0.95	0.56	1.45	1.33	1.95
B-Cell3	0.76	0.96	0.73	1.33	1.45	1.84
Colon	0.64	0.77	0.64	0.59	1.76	1.27
Embryonal Tumer	0.63	0.63	0.65	1.25	1.52	2.10
Lukemia1	0.60	0.92	0.58	1.57	1.73	2.58
Lukemia2	0.72	0.91	0.71	1.77	1.55	3.12
Pendigits	0.49	0.61	0.92	1.90	2.20	1.69
Synthetic	0.75	0.85	0.87	2.29	2.09	3.46
ECML	0.54	0.93	0.65	1.17	0.63	1.05
GCM	0.66	0.71	0.45	1.22	1.07	1.81
Diabetes	0.60	0.77	0.65	1.58	1.17	4.24
Glass	0.47	0.50	0.87	2.33	2.86	4.21

IGSC outperforms subspace clustering algorithm PROCLUS when both the number of clusters and number of features are high, such as for ECML and B_cell2 dataset. We find that purity measure of our proposed model IGSC is quite good for datasets with very high dimensions such as ECML and GCM. However, for low-dimensional dataset Pendigits and Glass, the information gain approach does not give very significant results as the contribution of attributes are equally likely in knowing the labels of instances.

5.3. Phase 2: experimental analysis of classification using Nearest neighbor approach

In this section, we present experimental results and comparison of our proposed modification to the traditional kNN algorithm. L1 data depth for individual clusters were plotted separately and the test instances that are similar to a cluster were compared with the data depth representation of that cluster. To understand how the proposed approach handles increase in dimension and number of instances, we organize the datasets as given in Table 4.

Table 4. Dataset used for phase 2 experimentation.

Dataset	High dimension	Low dimension
High records	• Bank • Synthetic	• German • Pendigits
Low records	• BCELL1 • Waveform	• Glass • Diabetes

5.3.1. Experimental procedure

The experiments were repeated 5 times and the average values were considered for final evaluation. We also performed 4-fold cross-validation for each dataset. The proposed approach was compared with traditional kNN and K-means+kNN classifier. The metrics used for validation purpose are accuracy of classification and F1 measure. To understand the change in accuracy for different k values (nearest neighbors), experiments were done by varying k from 5 to 25 and finally to entire depth points (full search) in the depth space.

5.3.2. Accuracy and F-measure

As explained in Algorithm 5, each test sample is classified only based on the best subspace cluster, S_b, to which it belongs to. kNN is applied on S_b by varying k from a low value to a maximum of $|S_b|$ value. We term $k = |S_b|$ as full search in Figures 5–8. For high-dimensional data, accuracy is comparable for various values of k as shown in Figure 5. From the results it can be noted that the intuition of depth does not bring a great change in the accuracy but classification time is low for high-dimensional data as shown in Figure 5. Since few datasets have uneven class distribution, we used F1 score (https://en.wikipedia.org/wiki/F1_score) also called F-measure as a performance measure of proposed approach. Here we calculated F-measure for each dataset by varying the values for k as shown in Figures 5–8. For each value of k, the higher the value of F-measure is, the better the classification. During full search, i.e. when $k = |S_b|$, the proposed approach has almost equal or higher F-measure value compared to kNN, for the given datasets. This indicates that our method is consistent as the F-measure does not vary much for different k values.

5.3.3. Classification time

The total time taken for classification was computed for both kNN and our approach. From the comparison we observed that classification time is consistently lesser than kNN except for datasets having low record size and low dimensions. During full search, our approach takes two orders of magnitude less time for classification than traditional kNN for high-dimensional and high record size of data as shown in Figure 5. For high record size and low-dimensional data the execution time is at least three orders of magnitude lesser than kNN as shown in Figure 6. For low records size and high-dimensional data, during full search, the execution time is at least four orders of magnitude lesser than kNN as shown in Figure 5.3.4. However, for low record size and low-dimensional data, the execution time is higher than kNN as shown in Figure 8 but lesser if k is a smaller value. Thus, the performance of proposed classification approach is at least two to three orders of magnitude better than conventional kNN approach, as the dimensions and record size increases.

5.3.4. Accuracy with change in Average number of dimensions per subspace cluster, l

To understand how the change in l affects accuracy, experiments were conducted by keeping k constant (selecting the best value from results of Section 5.3.2, as shown in Figures 5–8) and varying l for each dataset as shown in Table 5. The proposed method generates better or approximately the same results compared to traditional kNN for most of the datasets except for few cases like Pendigits and Waveform datasets. This is due to the fact that all the attributes in Pendigits and Waveform are important for a sample to be classified. Reducing the attributes lead to information loss and hence lead to lesser classification accuracy.

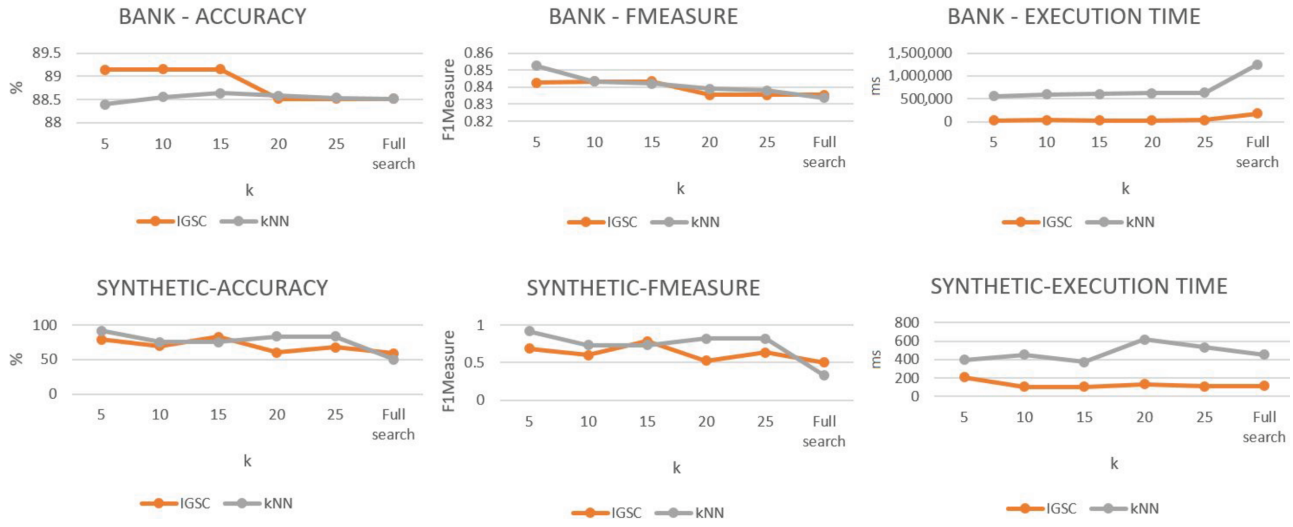


Figure 5. Comparison of the proposed approach with kNN. Dataset has high records and high dimension.

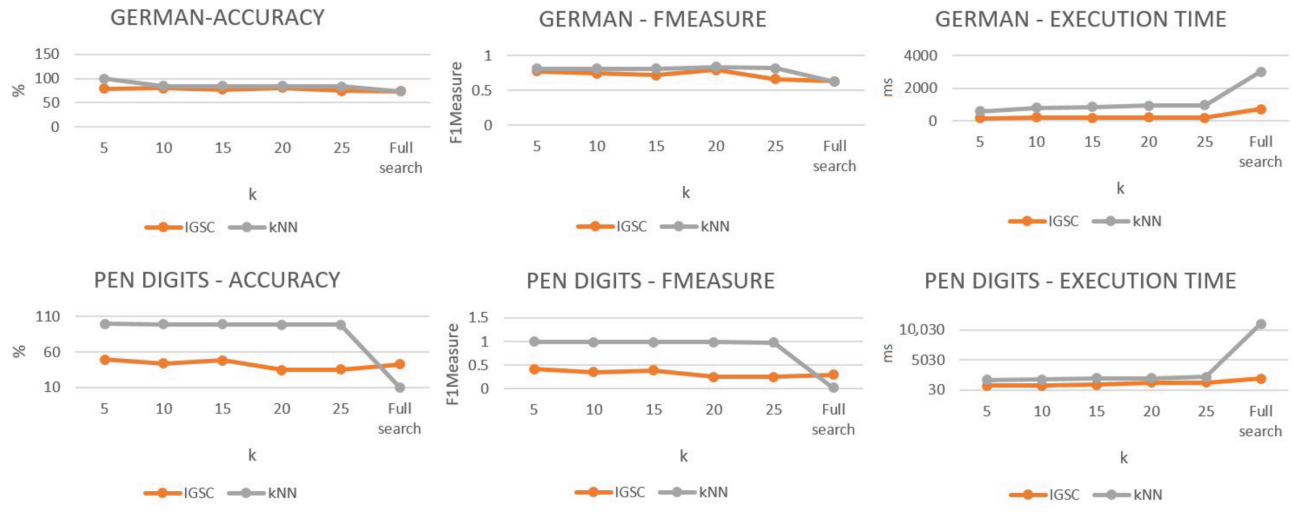


Figure 6. Comparison of proposed approach with kNN. Dataset has high records and low dimension.

5.4. Accuracy comparison

Table 6 shows the accuracy measure of our proposed approach for classification with kNN, K-means with kNN, and PCA with kNN. PCA with kNN first reduces the dimensionality of the entire dataset. Then kNN is

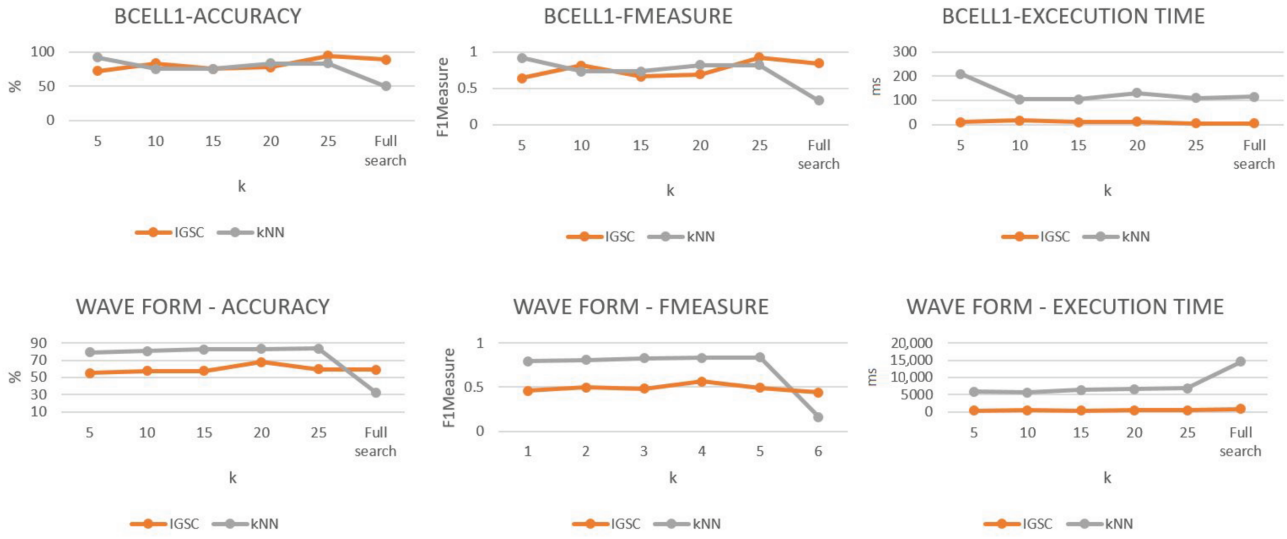


Figure 7. Comparison of the proposed approach with kNN. Dataset has low records and high dimension.

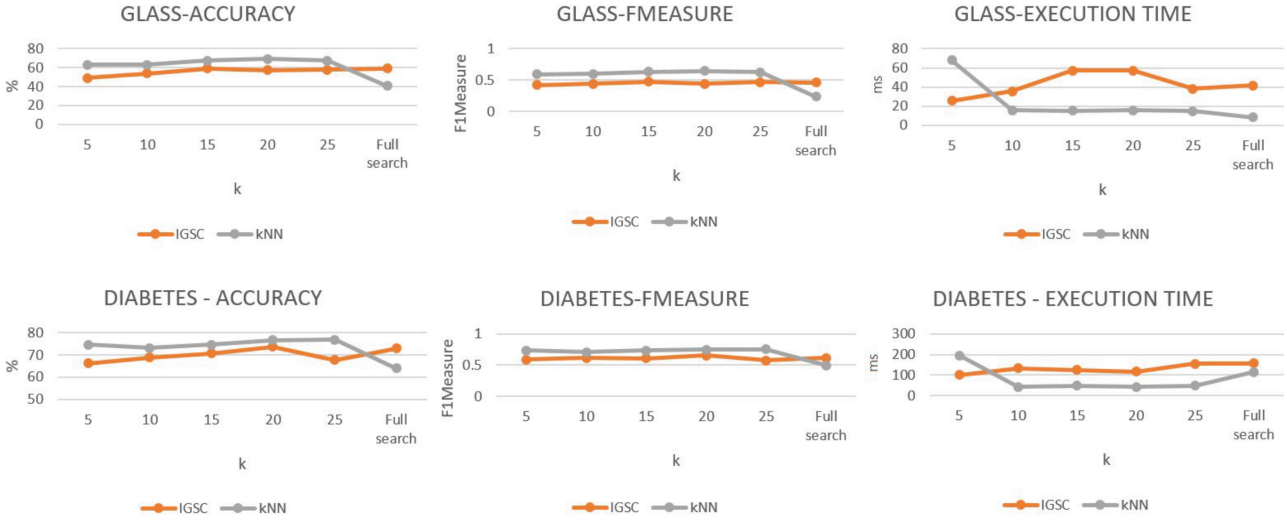


Figure 8. Comparison of the proposed approach with kNN. Dataset has low records and low dimensions.

Table 5. Classification accuracy of the proposed approach by varying average dimensions per cluster (l). For each dataset, in our proposed approach, k value is chosen as the one that gives best accuracy from experiment results in Section 5.3.2. k value in kNN is the one that gives highest accuracy amongst various values.

l dataset	A/3	A/4	A/5	A/6	kNN
Bank($k = 5$)	90.0679	93.43	96.587	96.60	90.8($k = 10$)
Diabetes($k = 30$)	74.5	73.7	76	76	71.3($k = 5$)
German($k = 5$)	75.42	75.42	75.42	75.42	71($k = 10$)
Pendigits($k = 5$)	53.815	62.1	59.464	50.496	99.9($k = 5$)
Waveform($k = 20$)	48.16	67.5	74.96	54.64	82($k = 10$)

applied on the reduced dataset. The dimensions of the reduced dataset is given by r for PCA+kNN in Table 6. Our proposed approach, IGSC-kNN outperforms other approaches in case of very high-dimensional data such as B-cell1, Leukemia1, Leukemia2, and Bank. For low records or low-dimensional dataset, there is no much variation in the accuracy. However, the execution time is quite less consistently for all datasets as can be seen from Figures 5 and 6. The proposed method generates better results compared to traditional kNN for most of the datasets except for few cases like Pendigits and Waveform datasets. This is due to the fact that all the attributes in Pendigits and Waveform are important for a sample to be classified. Reducing the attributes lead to information loss and hence lead to lesser classification accuracy. For Synthetic datasets, however, the results are comparable to kNN. PCA with kNN gives very good results for Synthetic, Tumer, and Diabetes datasets. This indicates that reducing the attributes in these datasets led to negligible loss in information and hence classifies a sample with better accuracy than kNN. PCA-kNN outperforms IGSC-kNN in these datasets as these datasets are better visualized in arbitrary oriented subspace and not in axis aligned subspace. As explained in Section 3, IGSC-kNN forms axis aligned subspaces but PCA-kNN reduces dimensionality in arbitrarily oriented space. Thus, for few datasets, arbitrarily oriented subspace gives the best results and not axis aligned subspaces.

Table 6. Comparison of accuracy measure with different datasets and different approaches.

Dataset	kNN (k)	Kmeans+kNN(K)	IGSC+kNN(K,k,l)	PCA+kNN(r,k)
German	71(10)	73(2)	74(5,5,6)	71.30(5,10)
Glass	59(10)	57(7)	61(5,10,2)	59.09(2,20)
Waveform	82(10)	83(15)	74.3(7,5,10)	46.31(6,10)
Bank	90.8(10)	90.4(9)	93.7(9,10,15)	90.24(15,10)
B-cell1	86(5)	72(2)	93.9(3,5,1006)	50.2(1000,10)
Colon	85.7(15)	88.8(5)	87.5(2,5,500)	86.23(500,10)
Diabetes	71.3(5)	73.56(5)	73.1(3,5,2)	77.27(2,20)
Tumer	70.83(15)	80.1(2)	78.05(2,5,1782)	88.88(2000,20)
Leukemia1	85.4(5)	88.8(2)	96.23(2,5,1782)	85.71(6,10)
Leukemia2	90.1(5)	92.3(5)	98.8(2,5,1782)	98.23(6,10)
PenDigits	99.9(5)	72.3(5)	60.55(9,5,4)	65.45(4,10)
Synthetic	88.64(10)	87.47(9)	87.47(9,5,18)	99.61(15,10)

6. Conclusion

We presented an approach to improve the nearest neighbor algorithm for high-dimensional data. A top-down approach of subspace clustering using information gain ranking for attribute extraction and representing the subspace clusters using L_1-depth for nearest neighbor classification is introduced as a new concept. This alternative attribute selection technique together with subspace extraction has been conducted in such a way that the data is neither transformed nor does it lose its structure. The feasibility of incorporating the concept of information theory for subspace selection and enhancing the nearest neighbor algorithm using L_1-depth for high-dimensional dataspace is established by the experimental results. The empirical results indicate that our approach generated comparable accuracy results with respect to traditional methods. We also observed that the classification time is consistently lesser than kNN except for few datasets such as Pendigits and Waveform. This is due to the fact that all the attributes play a significant role in classifying a sample in Pendigits and

Waveform datasets. During full search, our approach takes two orders of magnitude less time for classification than traditional kNN for high-dimensional and high record size of data as shown in Figure 5. The future scope lies in extending the approach to nonnumerical datasets in distributed environment. Existing results can be compared with a different approach of extracting relevant dimensions for each subspace cluster. This work can be extended theoretically to find the various subspace properties satisfied by the formed clusters and to find the approximate error from the Bayes classifier. Leveraging distributed computing to find such subspace clusters based on Entropy and Information gain still needs to be explored. The work can be extended to various application domains where the data is high-dimensional such as health-care for diagnostic analysis and finance for investment predictions.

Acknowledgment

We would like to thank the faculty of Computer Science and Engineering Department of Amrita Vishwa Vidyapeetham for their support.

Contribution of authors

Sandhya Harikumar conceived this idea and developed prototype. She devised algorithms and wrote the paper. A.S. Akhil did implementation and experimentation. M.R. Kaimal contributed in overall development of this idea, interpretation of the results, and gave suggestions on improving the paper.

References

[1] Murphy, Kevin P. Machine Learning: a Probabilistic Perspective. London, UK: MIT Press, 2013.

[2] Pandey S, Supriya M, Shrivastava A. Data classification using machine learning approach. In: Thampi SM, Mitra S, Mukhopadhyay J, Li, Kuan-Ching, James AP, Berretti S (editors). Intelligent Systems Technologies and Applications. Cham, Switzerland: Springer, 2017, pp. 112-122.

[3] Menon RR, Aswathi P. Document classification with hierarchically structured dictionaries. In: Thampi SM, Dasgupta S, Berretti S (editors). Intelligent Systems Technologies and Applications. Cham, Switzerland: Springer, 2016, pp. 387-397.

[4] Aggarwal CC. Data Classification: algorithms and applications, 1st ed. London, UK: Chapman & Hall/CRC, 2014.

[5] Oren Anava, Kfir Levy Y. k*-Nearest neighbors: From global to local. In: Proceedings of 30th Conference on Neural Information Processing Systems (NIPS); Barcelona, Spain; 2016.

[6] Cover T, Hart P. Nearest neighbor pattern classification. IEEE Transactions on Information Theory 2006; 13(1): 21-27.

[7] Song Y, Huang J, Zhou D, Zha H, Giles CL. IKNN: Informative k-nearest Neighbor Pattern Classification. Berlin, Germany: Springer Berlin Heidelberg, 2007, pp. 248-264.

[8] Koppen M. The curse of dimensionality. In: Proceedings of the 5th Online World Conference on Soft Computing in Industrial Applications (WSC5); 2000; pp. 4-8.

[9] Kriegel HP, Kroger P, Zimek A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Transactions on Knowledge Discovery from Data 2009; 3(1): 1-58.

[10] Kriegel HP, Kroger P, Zimek A. Subspace clustering. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2012; 2(4): 351–364.

[11] Parsons L, Haque E, Liu H. Subspace clustering for high-dimensional data: a review. SIGKDD Explor Newsl June 2004; 6(1): 90-105.

[12] Hund M, Sturm W, Schreck T, Ullrich T, Keim D et al. Analysis of patient groups and immunization results based on subspace clustering. In: Guo Y, Friston K, Aldo F, Hill S, Peng H (editors). Brain Informatics and Health. BIH, Lecture Notes in Computer Science. Cham, Switzerland: Springer, 2015, pp.358-368.

[13] Rakesh A, Johannes G, Dimitrios G, Prabhakar R. Automatic subspace clustering of high-dimensional data for data mining applications. In: Proceedings of the 1998 ACM SIGMOD international conference on Management of data; ACM Press; 1998; pp. 94-105.

[14] Kailing K, Kriegel HP, Kroger P. Density-connected subspace clustering for high dimensional data. In: proceedings of the 4th SIAM International Conference on Data Mining; Orlando, FL; 2004; pp. 46-257.

[15] Goil S, Nagesh H, Choudhary A. MAFIA: Efficient and scalable subspace clustering for very large data sets. Technical Report CPDC-TR-9906-010 Northwestern University, 1999.

[16] Aggarwal CC, Wolf JL, Yu PS, Procopiuc C, Park JS. Fast algorithms for projected clustering. In: Proceedings of the ACM SIGMOD international conference on Management of data; Philadelphia, PA, USA; 1999; pp. 61-72.

[17] Aggarwal CC, Yu PS. Finding generalized projected clusters in high-dimensional spaces. In: Proceedings of ACM SIGMOD international conference on Management of data; Dallas, TX, USA; 2000. pp. 70-81.

[18] Woo KG, Lee JH. FINDIT: A fast and intelligent subspace clustering algorithm using dimension voting. Information and Software Technology 2004; 46: 255-271.

[19] Harikumar S, Akhil AS. Semi supervised approach towards subspace clustering. Journal of Intelligent & Fuzzy Systems 2018; 34(3): 1619-1629.

[20] Yang J, Wang W, Wang H, Yu PS. δ-clusters: Capturing subspace correlation in a large dataset. In: International Conference on Data Engineering; San Jose, CA, USA; 2002. pp. 517-528.

[21] Vardi, Yehuda, Cun-Hui Zhang. The multivariate L1-median and associated data depth. In: Proceedings of the National Academy of Sciences of the United States of America 2000; 97(4): 1423-1426.

[22] Friedman JH, Bentley JL, Finkel RA. An algorithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software 2007; 3(3): 209-226.

[23] Gongde Guo, Hui Wang, Bell David A, Yaxin Bi, Kieran Greer. KNN model-based approach in classification. In: Meersman R, Tari Z, Schmidt DC (editors). On The Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE. OTM 2003. Lecture Notes in Computer Science, vol 2888. Berlin, Germany: Springer Berlin Heidelberg, 2003,pp.986-996.

[24] Parzen E. On estimation of a probability density function and mode. The Annals of Mathematical Statistics 1962; 33(3): 1065-1076.

[25] Ghaddar B, Naoum-Sawaya J. High dimensional data classification and feature selection using support vector machines. European Journal of Operational Research 2018; 265(3): 993-1004

[26] Zhongwen Z, Huanghuang G. Visualization study of high-dimensional data classification based on PCA-SVM. In: Proceedings of IEEE Second International Conference on Data Science in Cyberspace (DSC); Shenzhen; 2017. pp. 346-349.

[27] Hastie T, Tibshirani R, Friedman J. The elements of statistical learning. Springer Series in Statistics; New York, NY, USA: Springer New York Inc., 2001.

[28] Lan L, Shi H, Wang Z, Vucetic S. Active learning based on parzen window. In: Active Learning and Experimental Design workshop, in conjunction with AISTATS; Sardinia, Italy; 2010. pp. 99-112.

[29] Harikumar S, Haripriya H, Kaimal MR. Implementation of projected clustering based on SQL queries and UDFs in relational databases. In: Proceedings of IEEE Recent Advances in Intelligent Computational Systems (RAICS); Trivandrum; 2013. pp. 7-12.

[30] Langley P, Blum AL. Selection of relevant features and examples in machine learning. Artificial Intelligence 1997; 97(1-2): 245-271.

[31] Ayan NF. Using information gain as feature weight. In: TAINN'99 8th Turkish Symposium on Artificial Intelligence and Neural Networks; Istanbul; 1999; pp. 48-57.

[32] Modha DS, Spangler WS. Feature weighting in k-means clustering. Machine Learning 2003; 52(3): 217-237.

[33] Gan GJ, Wu JH. A convergence theorem for the fuzzy subspace clustering (FSC) algorithm. Pattern Recognition 2008; 41(6): 1939-1947.

[34] John GH, Kohavi R, Pfleger P. Irrelevant features and the subset selection problem. In : Eleventh International Conference on Machine Learning (ICML);New Brunswick, NJ, USA; 1994. pp. 121-129.

[35] Vencalek O. Weighted data depth and depth based discrimination. PhD, Charles University, Prague, The Czech Republic, 2011.

[36] Vencalek O. New depth-based modification of the k-nearest neighbor method. SOP Transactions on Statistics and Analysis 2013; 1(2): 131-138.

[37] Hlubinka D and Vencalek O. Depth-based classification for distributions with nonconvex support. Journal of Probability and Statistics 2013. doi:10.1155/2013/629184

[38] Lange T, Mosler K, Mozharovskyi P. Fast nonparametric classification based on data depth. Statistical Papers 2014; 55(1):49-69.

[39] Vencalek O. Depth-based classification for multivariate data. Austrian Journal of Statistics 2017; 46(3-4): 117-128.

[40] Pokotylo O, Mozharovskyi P, Dyckerhoff R. Depth and depth-based classification with R-package ddalpha. arXiv preprint arXiv:1608.04109,2016.

[41] Dutta S, Sarkar S, Ghosh AK. Multi-scale classification using localized spatial depth. Journal of Machine Learning Research 2016; 17(217):1-30.

[42] Serfling R, Zuo Y. General notions of statistical depth function. Annals of Statistics 2000; 28: 461-482.

[43] Quinlan JR. Induction of decision trees. Machine Learning 1986; 1(1): 81-106.

[44] Lopez-Pintado S, Romo J. On the concept of depth for functional data. Journal of the American Statistical Association 2009; 104(486): 718-734.

[45] Davies DL, Bouldin DW. A cluster separation measure. IEEE Transactions on Pattern Analysis Machine Intelligence 1979; 1(2): 224-227.