Applied Mathematics & Information Sciences

Volume 17
Issue 5 *Sep. 2023*

Article 10

9-1-2023

Compatibility of Clique Clustering Algorithm with Dimensionality Reduction

Ug˘ur Madran
College of Engineering and Technology, American University of the Middle East, 54200 Egaila, Kuwait, Ugur.Madran@aum.edu.kw

Duygu Soyog˘lu
College of Engineering and Technology, American University of the Middle East, 54200 Egaila, Kuwait, Ugur.Madran@aum.edu.kw

Recommended Citation
˘ur Madran, Ug and Soyog˘lu, Duygu (2023) "Compatibility of Clique Clustering Algorithm with Dimensionality Reduction," *Applied Mathematics & Information Sciences*: Vol. 17: Iss. 5, Article 10.
DOI: http://dx.doi.org/10.18576/amis/170510
Available at: https://digitalcommons.aaru.edu.jo/amis/vol17/iss5/10

Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/170510

Compatibility of Clique Clustering Algorithm with Dimensionality Reduction

Uğur Madran[*] *and Duygu Soyoğlu*

College of Engineering and Technology, American University of the Middle East, 54200 Egaila, Kuwait

Received: 14 Jan. 2023, Revised: 20 Jul. 2023, Accepted: 21 Aug. 2023
Published online: 1 Sep. 2023

Abstract: In our previous work, we introduced a clustering algorithm based on clique formation. Cliques, the obtained clusters, are constructed by choosing the most dense complete subgraphs by using similarity values between instances. The clique algorithm successfully reduces the number of instances in a data set without substantially changing the accuracy rate. In this current work, we focused on reducing the number of features. For this purpose, the effect of the clique clustering algorithm on dimensionality reduction has been analyzed. We propose a novel algorithm for support vector machine classification by combining these two techniques and applying different strategies by differentiating the clique structures. The results obtained from well known data sets confirm the compatibility of clique clustering algorithm with dimensionality reduction.

Keywords: clique clustering, classification, dimensionality reduction, machine learning, support vector machine

1 Introduction

Machine Learning (ML) is one of the main stream research topics and can be thought as a collection of algorithms which allows one to have predictions without explicitly coding. Ideally, ML is used by running algorithms (so-called “machines”) to learn automatically from observed data (so-called “training data”) and to predict the outcomes of future decisions. The performance of algorithms is generally measured by their accuracy rates to predict unseen data (the so-called “test data”). ML algorithms have wide range of applications, such as image processing (e.g., fMRI scans, face detection), speech recognition (e.g., dictation, transcription), text categorization (e.g., email filtering), health sciences, engineering, social networks, business operation patterns, financial markets and many others (e.g., see [1], [2], [3]).

ML is generally divided into three categories; *unsupervised, supervised and reinforcement learning* (e.g., clustering, classification, and dynamic programming, respectively). In the last decade, a blended approach was observed as *semi-supervised learning* (see [4]) which is a union of supervised learning and unsupervised learning. Similarity-based approaches have been the most popular research topics in recent years among semisupervised learning techniques (see [5], [6], [7] and references therein).

In our previous paper [7], we introduced an algorithm which uses a similarity function to reduce the size of the data set by clustering the data based on “clique” formation. The cliques are being used in random matrix theory, and in recent years, this concept is also applied in clustering algorithms (for example, [8], [9], [10], [11]). For more details, we direct the reader to our original article [7], and brief information on the clique clustering algorithm will be given in Section 2.

Machine Learning algorithms for supervised learning are generally divided into two broad categories based on their purpose; classification ([12]) and regression ([13]). Classification algorithms predict a class label among the categorical set of labels, whereas regression algorithms predict a class value which is assumed to be a continuous data. As in our previous paper, we will focus on classification algorithms, and Support Vector Machines will be used.

Support vector machine (SVM) is a classification algorithm which is used with great success in various fields ([14], [15], [12]). Improving SVMs with different strategies is a very active research topic; it is possible to use SVMs on pre-processed data after using clustering algorithms (see, for example, [16], [17], [18], [19], [20],

[*] Corresponding author e-mail: Ugur.Madran@aum.edu.kw

[21], [22]). The common idea of all these approaches is to obtain comparable performance with reduced training data set size and with less number of support vectors. A short introduction to SVMs is also provided in Section 3.

The aim of this paper is to investigate the compatibility of our clique clustering algorithm with the most common dimensionality reduction method, known as recursive feature elimination (RFE). Generally, in many applications, data is provided in tabular form. The clique clustering algorithm provides a reduction in the number of rows of this data, and the dimensionality reduction algorithms provide a reduction in the number of columns (see Section 4). Therefore, the combined effect will reduce the data from both sides, with accuracy rates comparable to those of the machine learning algorithms as of the original data set, providing a more efficient model.

We tested our new method, on well-known data sets and provided the analysis of results. The results confirm the compatibility of clique-clustering algorithm with dimensionality reduction techniques. In order to obtain comparative results, 4 data sets from our previous paper are used, which have sufficiently many columns, and 4 new data sets are included to provide sufficient evidence. A discussion of the results can be found in Section 5.

2 Clique Clustering

Clustering is mainly a type of unsupervised machine learning technique, and is used to reveal patterns in the given data set, mainly by grouping *similar* observations. One of the well-known such algorithm is k-Nearest Neighbours. In semi-supervised ML algorithms, clustering techniques are combined with various supervised ML algorithms to improve the performance.

The term “cluster” does not have a precise definition, and there are different techniques to generate clusters that lead to different algorithms. The common idea for all of them is to group (or categorize) the data based on some measures of similarity or distance. In our previous paper, we showed that the clique clustering technique is an effective way to obtain clusters.

The *clique*, is defined as a complete subgraph (of an undirected graph), and in our approach, cliques are calculated from the associated graph of the data where the vertices denote the instances and the edges denote the similarities between instances. Then, by using a threshold value, we obtain the subgraph on which cliques are formed with an heuristic approach. We direct the reader to [7] for details and for an example calculation with visualizations. Here, we include the algorithm for the convenience of the reader. For the algorithm, t denotes a chosen threshold value, $sim(x_i, x_j)$ denotes a similarity function between the data points x_i and x_j, and B denotes the set of index pairs (i, j) for which $sim(x_i, x_j) \geq t$.

2.1 Clique Clustering Algorithm

Input: $B, t, sim(x_i, x_j)$.
Output: List of Cliques, L.

```
I.Let $B$ denote the ordered list of all pairs $(i,j)$ with $sim(x_i,x_j) \geq t$, for a given threshold $t$, and this list is ordered by the similarity values from largest to smallest.
II.Set $L := [\,]$ and $B_0 := B$ for initialization.
   Repeat until $B_0 = \emptyset$ :
      (i)Let $(i,j) \in B_0$ be the first element (i.e., the pair with the highest similarity value).
      (ii)Consider the sets $A_i = \{k \mid (i,k) \in B_0$ or $(k,i) \in B_0\}$ and $A_j$ similarly, and then form their intersection for possible elements of the clique; $A = A_i \cap A_j$. Sort $A$ as an ordered list by similarity values from largest to smallest according to $sim(x_i,x_k)$ for $i < j$ and $k \in A$.
      (iii)Set $A_0 := A$ for initialization
         Repeat until $A_0 = \emptyset$
         –Choose $k \in A_0$ as the first element (i.e., with the largest similarity value of $sim(x_i,x_k)$).
         –$A_0 := A_0 \backslash \{k\}$.
         –$A_k := \{\ell \in A \mid (k,\ell) \in B_0$ or $(\ell,k) \in B_0\}$.
         –Compare (as sets) $A \cup \{i,j\}$ and $A_k \cup \{k\}$, and find any missing elements, if any:
          $D := \big(A \cup \{i,j\}\big) \backslash \big(A_k \cup \{k\}\big)$
         –Update $A$ and $A_0$ as $A := A \backslash D$ and $A_0 := A_0 \backslash D$.
         –Next.
      (iv)$L := \text{Append}(L,[A])$ update the list $L$.
         A clique is obtained, $A$, and added to the list of cliques. Now, we will remove from $B_0$ any pair associated with clique vertices.
      (v)For each $k \in A$, set
         $R_k = \{(k,\ell) \mid (k,\ell) \in B_0\} \cup \{(\ell,k) \mid (\ell,k) \in B_0\}$.
      (vi)$R = \cup_{k \in A} R_k$.
      (vii)$B_0 := B_0 \backslash R$.
   (viii)Next.
III.Return $L$.
```

We distinguish two types of clique: homogeneous cliques and heterogeneous ones. For an homogeneous clique type, we consider the ones consisting of observations belonging to the same class label. Similarly, if a clique includes observations from more than one class label, it is considered as an heterogeneous clique type.

After the cliques are obtained, we reduce the number of observations in the data set before using any classification algorithms. The idea of reducing the number of data is to reduce the computation time required by the machine learning algorithm to process the training data. For this purpose, we previously introduced three different strategies; Centers of Cliques-SVM (CC-SVM), Homogeneous Cliques Removed-SVM (HOM-R-SVM), and Heterogeneous Cliques Removed-SVM (HET-R-SVM). As before, in this note, we will use only

SVM as a classification algorithm with numerical data. These three strategies are also summarized below for the reader's convenience.

–*Centers of Cliques-SVM (CC-SVM) Algorithm*:
The centers of all instances within the clique are calculated for each class. For a homogeneous clique, instead of taking all the data, we consider only its center as one observation. For the non-homogeneous (i.e., heterogeneous) one, the number of data is reduced to the number of the class labels observed within the clique for training.

–*Homogeneous Cliques Removed-SVM (HOM-R-SVM) Algorithm*:
Before using the classification algorithm (e.g., SVM), we remove all homogeneous cliques. Therefore, only heterogeneous cliques will be considered, and their centers will be used for the classification. If the aim of using the data is to distinguish the heterogeneous types (for example, to detect anomalies), then this algorithm might be more useful.

–*Heterogeneous Cliques Removed-SVM (HET-R-SVM) Algorithm*:
This is the opposite of the HOM-R-SVM Algorithm; we remove all heterogeneous cliques before training the SVM and only homogeneous cliques are considered. This method is more useful if cliques in the data are supposed to be well separated by class labels, and the heterogeneity is due to noise or data errors.

3 Support Vector Machines

Support Vector Machine (SVM) is an algorithm used for classification in ML [23]. For ML algorithms, the data are first divided into test and training splits. Then, using the training data set, SVM builds hyperplanes to separate the data into categories (mostly called classes). SVMs were developed by Vapnik and Chervenenkis in 1974 (see [24]) and are still the most popular ML algorithms for classification. For recent developments and improvements on SVMs see [25], [26], [27], and references therein.

To achieve data classification, SVM aims to find the best hyperplane by maximizing the distance (called margin) between the hyperplane and the closest data points from each class.

When the data set is n dimensional, the optimization problem can be given as:

$$\text{Minimize } \|\boldsymbol{w}\| \text{ subject to } y_i(\boldsymbol{x_i}\cdot\boldsymbol{w}-b) \geq 1 \text{ for } i=1,2,\ldots,k$$

where $\boldsymbol{w}$ is a normal vector to the hyperplane, y_i is either -1 or 1 each indicating the class to which the point $\boldsymbol{x}_i$ belongs to. Here, the equation of the hyperplane is assumed to be given as $\boldsymbol{x}\cdot\boldsymbol{w}+b=0$, explaining the constant coefficient b. In the articles [12,28], one may find details for a comprehensive and complete account of the underlying convex optimization.

Another advantage of the SVM is its applicability even when the data is not linearly separated. For this, a kernel function $K(\boldsymbol{x_i},\boldsymbol{x_j})$ can be used to transform the data into a linearly separable space, that is, the data are transformed into a higher-dimensional Euclidean space. In Table 1, the most commonly used kernel functions are listed.

Table 1 Common kernel functions

Linear:	$K(\boldsymbol{x_i},\boldsymbol{x_j})=\langle\boldsymbol{x_i},\boldsymbol{x_j}\rangle$
Polynomial:	$K(\boldsymbol{x_i},\boldsymbol{x_j})=(\gamma\langle\boldsymbol{x_i},\boldsymbol{x_j}\rangle+r)^d$
Gaussian Radial Basis (rbf):	$K(\boldsymbol{x_i},\boldsymbol{x_j})=e^{-\gamma\|\|\boldsymbol{x_i}-\boldsymbol{x_j}\|\|^2}$
Sigmoid:	$K(\boldsymbol{x_i},\boldsymbol{x_j})=\tanh(\gamma\langle\boldsymbol{x_i},\boldsymbol{x_j}\rangle+r)$

In our study, all of the kernel functions listed in this table are used, and the accuracies of the resulting SVMs are calculated to find the best kernel function representing the data (indeed, to find the best separating hypersurface). In real life applications, the parameters of these kernel functions (also called hyper-parameters) can be fine tuned for better performance. The default parameters for the kernel functions are defined as; $\gamma=1/(n\ s^2), r=0, d=3$, where $\boldsymbol{x_i}\in\mathbb{R}^n$ and s^2 denotes the variance of the set $\{\boldsymbol{x_1},\ldots,\boldsymbol{x_k}\}$.

4 Dimensionality Reduction

In machine learning, similar to other problems with sampling and modeling, in order to obtain better results, it is common to start with higher-dimensional data and then remove irrelevant features ([29,30]). When considering more dimensions, to be able to obtain reliable results, the number of observations should also be increased exponentially (with respect to the number of features observed). This brings, other than the increase in the size of data, an undesired effect which causes the inefficiency; requiring more computation time and more power. Moreover, insignificant features contribute not much on the performance of the model. This behavior is commonly known as the "*curse of dimensionality*".

To remedy this problem, several techniques are introduced in the literature, such as "*low variance filter*", "*forward feature selection*", "*backward feature elimination*", "*factor analysis*", "*principal component analysis (PCA)*", "*independent component analysis*", "*random forest*", "*t-SNE*", "*UMAP*", "*ISOMAP*". A comparative review of some dimentionality reduction algorithms and their limitations were studied in [31], [32]. Another aspect of dimensionality reduction is the visualization and interpretation of the model.

In this study, our aim is to investigate the compatibility of the clique-clustering algorithm with a dimensionality reduction method. For this reason, we will focus on the most commonly used technique; "Recursive Feature Elimination" (RFE) which is also referred in the literature as backward feature elimination. That is, we will combine the clique-clustering algorithm with RFE by reducing the number of features, and then use SVM as a classification. Hence, the data set will be reduced not only with the number of instances but also with the number of features to reduce the computation time, without sacrificing much on the performance of the model.

4.1 RFE

Recursive Feature Elimination (RFE) is one of the most popular and successful feature selection algorithm. In this algorithm, first, the model is trained by considering all the features given in the data set. After training the model, the RFE algorithm removes the least important feature considering the performance of the model with the remaining features on the test data. This process is repeated, until the desired number of features remains. This can be done by eliminating one feature at a time or more features at each iteration, depending on the application and the needs. In this paper, we used the default behaviour; eliminating one feature at a time.

To correctly identify the importance values of the features, the features must be scaled. This is achieved by transforming them to their standardized values, i.e., $z_i = \frac{x_i - \bar{x}}{s}$. This is implemented in the code with the `StandardScaler()` from the `preprocessing` module in the `scikit-learn` library. The reason for considering standardized values, instead of other scaling methods (such as `normalize`, or `MinMaxScaler`), is to obtain a more robust analysis in which outliers should have a small effect on performance.

4.2 Implementation

This section includes a brief overview of the proposed methods and their parameters. The results will be given in the next section (Section 5) by comparing the performance of each proposed algorithm.

For this current study, we ported our model to Python and used the scikit-learn library (ver. 1.0.2 [33]), our previous work had been run on R-Software with ‘e1071’ package. All data sets are standardized by StandardScaler() and divided into “training” and “test” splits with test size equal to 30%, and with stratification of the target variable, i.e., class labels. Moreover, for each data set, we run 5 random splits and the average scores are reported. For all SVM runs, the default parameters are used so that the results can be easily compared with the previous ones without any bias.

Moreover, for each data set, different threshold values (80%, 85%, 90%, 95%) are considered for clique formations. With $t = 80\%$, there are more cliques formed for most of the data sets, and the data size is reduced almost to tenth of the original data. With only a few number of data remaining, the performance of SVM is significantly dropped. Depending on the characteristics of the data sets, other threshold values reach comparable performance with the original SVM with reduced data size. The other extreme value, $t = 95\%$, limits the clique formation for most of the data sets, and hence, the reduction of the data size is significantly limited.

Also, for each data set, all possible SVM kernels mentioned in Table 1 are considered and the best performing kernel for the standard SVM is chosen as a reference and it is used for all other algorithms for comparison (see below tables). Since, the last 4 data sets are new to clique clustering algorithms, here we provide the accuracy scores of each algorithm (see Tables 2, 3, 4), for the convenience of the reader. We also include the results for the other 4 data sets for completeness.

Table 2 Accuracy scores of CC-SVM vs Std. SVM

Data Set	t	ker	n	n_{red}	SVM	CC
Wine	0.95	rbf	124	117.8	0.9963	0.9926
Sonar	0.95	rbf	145	139.2	0.8317	0.8286
Ionoshp.	0.95	rbf	245	163.4	0.9472	0.9396
Vehicle	0.95	rbf	592	401	0.7614	0.7504
Divorce	0.9	poly	119	61.6	0.9882	0.9725
Musk	0.95	rbf	333	251	0.9133	0.9007
Parkins.	0.95	rbf	168	145.8	0.8333	0.8028
Spect	0.8	rbf	186	87	0.8469	0.7901

Table 3 Accuracy scores of HET-R-SVM vs Std. SVM

Data Set	t	ker	n	n_{red}	SVM	HET-R
Wine	0.95	rbf	124	117.8	0.9963	0.9963
Sonar	0.95	rbf	145	139.2	0.8317	0.8286
Ionoshp.	0.95	rbf	245	160.2	0.9472	0.9396
Vehicle	0.95	rbf	592	237.8	0.7614	0.7024
Divorce	0.9	poly	119	61.6	0.9882	0.9725
Musk	0.95	rbf	333	225.4	0.9133	0.8685
Parkins.	0.95	rbf	168	39.6	0.8333	0.7944
Spect	0.8	rbf	186	66.6	0.8469	0.7901

It must be noted that, for an individual proposed algorithm with clique clustering, better accuracy scores and more reduced data sets are also observed by using smaller threshold values. For easy comparison, we include here the same threshold values for each algorithm for the same data set. An example is better to see this comparison and allows us explain why the discrepancy happens here, mainly due to high number of cliques being removed.

As can be seen in Table 5 for the Wine data set, with the CC-SVM algorithm, $t = 0.80$ provides sufficient

Table 4 Accuracy scores of HOM-R-SVM vs Std. SVM

Data Set	t	ker	n	n_{red}	SVM	HOM-R
Wine	0.95	rbf	124	111.6	0.9963	0.9926
Sonar	0.95	rbf	145	133.8	0.8317	0.8254
Ionoshp.	0.95	rbf	245	134.8	0.9472	0.9415
Vehicle	0.95	rbf	592	318.6	0.7614	0.6953
Divorce	0.9	poly	119	39	0.9882	0.9686
Musk	0.95	rbf	333	192.6	0.9133	0.8140
Parkins.	0.95	rbf	168	131.6	0.8333	0.8000
Spect	0.8	rbf	186	62	0.8469	0.7877

performance with only 17 clique centers (n_{cen}) and more accurate rates with $t = 0.90$ with cliques of half size of the data. Here, n_{red} (i.e., $n_{\text{cen}}, n_{\text{hom}}, n_{\text{het}}$, respectively for each algorithm) denotes the average number of instances used to train SVM, and rel$_{\text{score}}$ denotes the ratio of the performance of the given algorithm to the performance of the standard SVM when full data set is used. On the other hand, the cliques formed for $t = 0.80$ are mostly homogeneous ones, and for the HOM-R-SVM algorithm only 9 clique centers (n_{hom}) remains, which is clearly insufficient to get reliable conclusions.

Based on our experiments, we recommend as a rule of thumb that, threshold value should be chosen so that the number of centers obtained should be about half the size of the original data. For the Wine data set, $t = 0.90$ will be enough for CC-SVM and HET-R-SVM, but a higher value is recommended for HOM-R-SVM. Indeed, $t = 0.925$ gives the optimum results with $n_{\text{red}} = 65$ and rel$_{\text{score}} = 0.9777$.

4.3 RFE algorithms

For each data set, first, standard SVM and clique related algorithms are run to obtain the reference performance values. Moreover, after this initial run, the importance values of each feature is calculated. To obtain reliable results, each feature column is scaled with standardized scores.

- *RFE-CC-SVM:*

Step 1: Apply the clique clustering algorithm (see Section 2).
Step 2: Obtain the centers of the cliques.
Step 3: Standardize the values of all data to identify the most important features.
Step 4: Using the given ratio for RFE, select the most important features, recursively.
Step 5: Apply SVM with selected features and using only centers of cliques, together with instances that do not belong to any clique.

- *RFE-HET-R-SVM:*

Step 1: Apply the clique clustering algorithm.
Step 2: Obtain the centers of the cliques after removing all heterogeneous cliques.
Step 3: Transform the centers of the cliques to their standardized scores to find the most important features.
Step 4: By using the given ratio for RFE, select the features, recursively.
Step 5: Apply SVM with selected components and using only centers of homogeneous cliques together with instances that do not belong to any clique.

- *RFE-HOM-R-SVM:*

Step 1: Apply the clique clustering algorithm.
Step 2: Get the centers of the cliques after removing all homogeneous cliques. Note that, for each clique, there will be at least 2 centers, one for each class label appearing in the clique.
Step 3: Obtain the standardized scores of the centers of cliques.
Step 4: By using the given ratio for RFE, select the features, recursively.
Step 5: Apply SVM with selected components and using only centers of heterogeneous cliques together with the instances that do not belong to any clique.

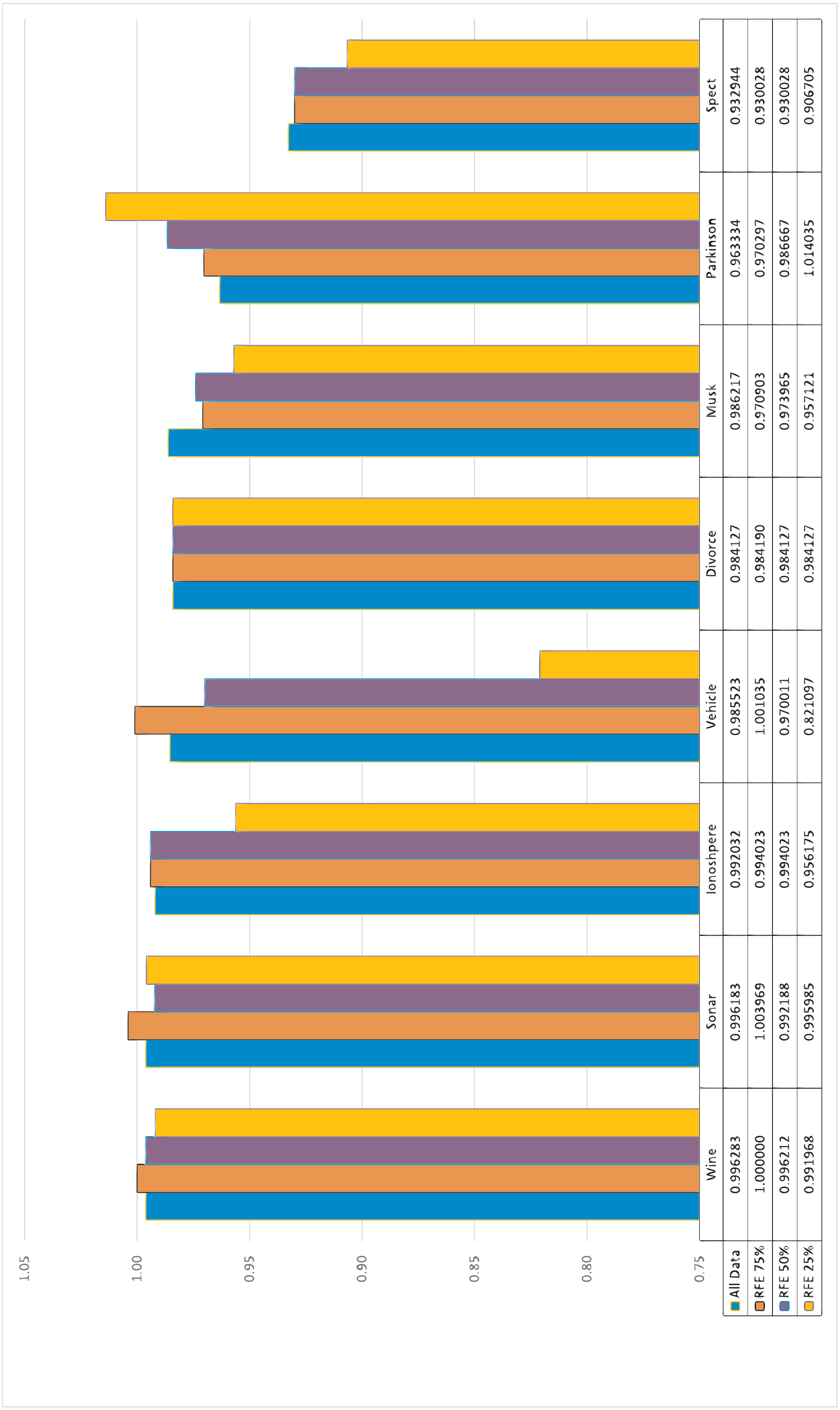
The results are given as bar graphs in Section 5 by comparing the performances of RFE-75%, RFE-50%, and RFE-25% within each algorithm. Note that the values 75%, 50%, and 25% are generic values chosen to test the compatibility of clique clustering algorithms, and specific values can be used for different data sets, depending on their structure and the needs for the ML task.

5 Results and Discussion

In order to analyze the performance of the proposed methods we consider 8 publicly available data sets from UCI database. The first four data sets have also been used in our previous study (the ones chosen here are those with more than 10 attributes) and are included here for comparative results ([7]). Moreover, 4 new data sets have been studied, having attribute sizes varying between 23 and 167, as listed below. As mentioned earlier, all data sets are chosen so that each has numerical attributes only. The data sets also represent different areas of life with various properties; for example, Spect data set consists of only binary data. Divorce data set consists of questionnaire responses scaled from 0 to 4 (integer values).

For the “Ionosphere” data set, the second attribute is constant for all instances, so this attribute is completely removed before using the data. For the “Musk” data set, `clean1.data` is used, as summarized above. For the “Spect” data set, the files `SPECT.train` and `SPECT.test` are merged and random splits are formed.

It has been observed that RFE-CC-SVM-50% performs quite well, on average, with all data sets considered. As expected, the RFE-25% method is quite unstable compared to the larger values. Nevertheless, it



	Wine	Sonar	Ionoshpere	Vehicle	Divorce	Musk	Parkinson	Spect
🟦 All Data	0.996283	0.996183	0.992032	0.985523	0.984127	0.986217	0.963334	0.932944
🟧 RFE 75%	1.000000	1.003969	0.994023	1.001035	0.984190	0.970903	0.970297	0.930028
🟪 RFE 50%	0.996212	0.992188	0.994023	0.970011	0.984127	0.973965	0.986667	0.930028
🟨 RFE 25%	0.991968	0.995985	0.956175	0.821097	0.984127	0.957121	1.014035	0.906705

Fig. 1 Relative performance rates of RFE-CC-SVM

Table 5 An example for the effect of different threshold values.

Data	ker	t	n	SVM	n_{cen}	rel$_{\text{score}}$	n_{het}	rel$_{\text{score}}$	n_{hom}	rel$_{\text{score}}$
Wine	rbf	80	124	0.9963	16.6	0.9517	11.4	0.8513	8.8	0.5651
Wine	rbf	85	124	0.9963	28.6	0.9628	25	0.9554	10.2	0.5056
Wine	rbf	90	124	0.9963	61.4	0.9926	60.2	0.9926	31.2	0.9033
Wine	rbf	95 *	124	0.9963	117.8	0.9963	117.8	0.9963	111.6	0.9963

Table 6 Number of Features used with RFE

Data Set	a_{All}	$a_{75\%}$	$a_{50\%}$	$a_{25\%}$
Wine	12	9	6	3
Connectionist Bench (Sonar_All)	60	45	30	15
Statlog (Vehicle)	18	13	9	4
Ionosphere	33	24	16	8
Divorce	54	40	27	13
Musk	166	124	83	41
Parkinson	45	33	22	11
Spect	22	16	11	5

Table 7 Summary of Data Sets

Data Set	n	a	c	Area
Wine	178	12	3	Physical
Connectionist Bench (Sonar_All)	208	60	2	Physical
Statlog (Vehicle)	846	18	4	Physical
Ionosphere	351	33	2	Physical
Divorce	170	54	2	Life
Musk	476	166	2	Physical
Parkinson	239	45	2	Life
Spect	267	22	2	Life

should be mentioned that, the scale of the bar graphs are set for their performances between 0.75 and 1.05 to ensure the visibility of the comparisons. The minimum performance is observed for the Vehicle data set with 82% at RFE-25%. For RFE-50%, the minimum performance is observed with the Spect data set with 93% relative performance, still a very good performance with half of the features. It should also be noted that for the Spect data set, $t = 0.80$ was used and $\frac{n_{\text{red}}}{n} = \frac{87}{186} = 0.4677$, with RFE-50% the ratio of all the data used is reduced to 0.2339. In summary, with only 23.29% of the original data, we received 93% of the corresponding performance.

The second most successful model proposed is the RFE-HET-R-SVM algorithm based on the average performance rates for the given data sets. Indeed, the performance of RFE-HET-R-SVM is the same as the performance of RFE-CC-SVM for 2 data sets (Wine and Sonar), slightly better for 3 data sets (Ionosphere, Divorce, and Spect) and slightly worse for the remaining 3 data sets (Vehicle, Musk, and Parkinson). It is also interesting to note that, for the Divorce data set, both RFE-HOM-R-SVM and RFE-CC-SVM give the same performance ratios for all RFE parameters, and only RFE-HET-R-SVM gives different performance ratios.

Even though RFE-HOM-R-SVM performs with the least average relative score, for 3 data sets (Wine, Ionosphere, and Spect) it performs slightly better than RFE-CC-SVM and RFE-HET-R-SVM at 50% parameter. Moreover, it should also be noted that, among all the algorithms proposed, the CC-SVM algorithm has the lowest number of training data, providing the best reduction in the size of the training data.

5.1 *Closing remarks and future work*

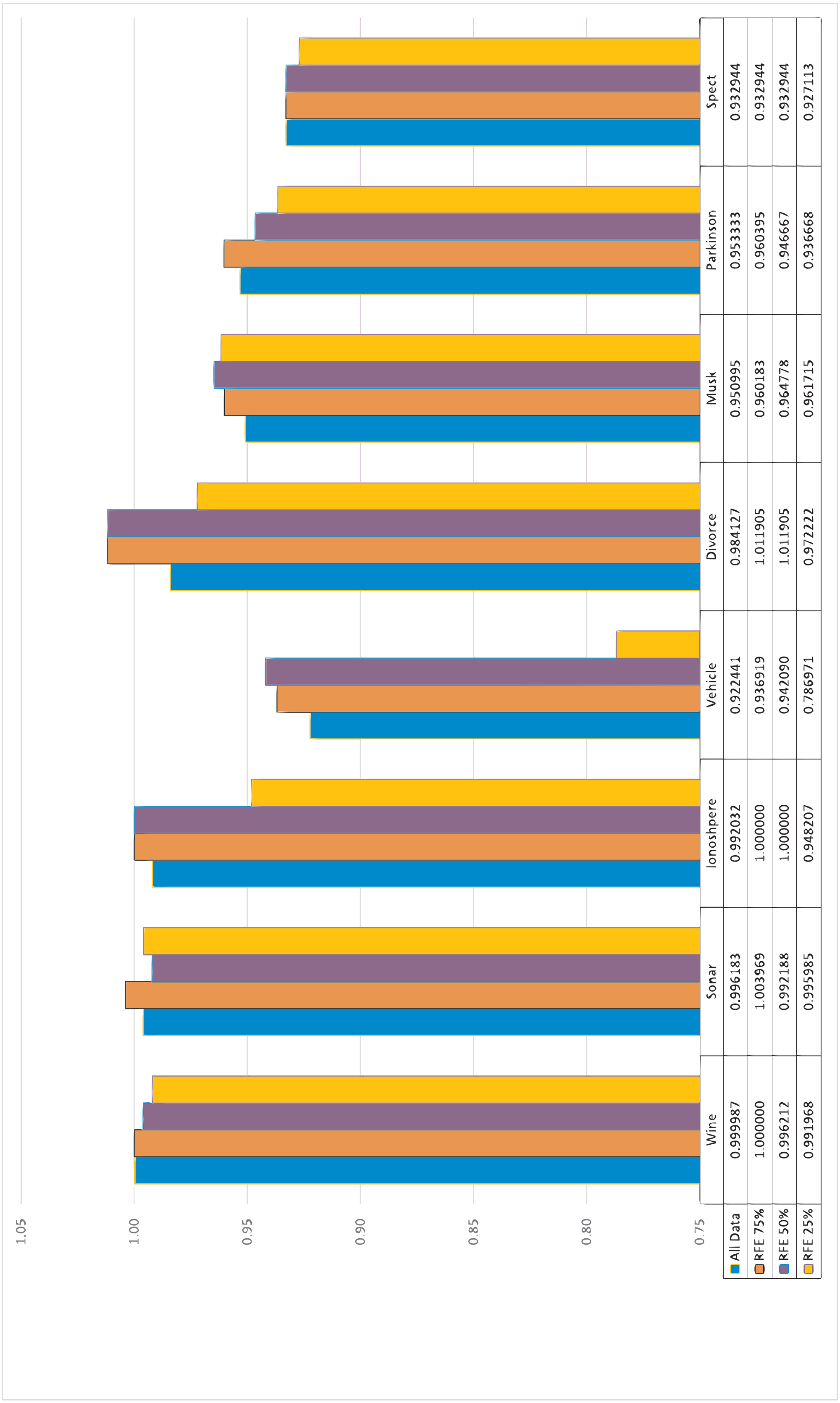
In conclusion, based on the results represented in this note, it has been showed that the clique clustering algorithms are compatible with RFE dimensionality reduction technique. It is worth further analyzing the other dimensionality reduction techniques, such as PCA and also other ML algorithms, such as Random Forest. Another direction to expand this study could be to analyze the performance of the clique clustering algorithm after fine-tuning the parameters of SVMs (or other ML algorithms). That is, the clique-clustering algorithm can be used after hyper-parameter tuning with the best parameters obtained for each data set.

Furthermore, in our previous study and also in this study, we focused only on the classification problem. It is also interesting to analyze the performance of the clique clustering algorithms for regression problems.

For future work, we would like to expand these studies to larger data sets by partially using clique clustering on the smaller subsets and use them repeatedly until data reduction is achieved to an acceptable level. If this is achieved, then the computation time of the cliques will be substantially reduced, and this method can be used effectively with many machine learning algorithms.

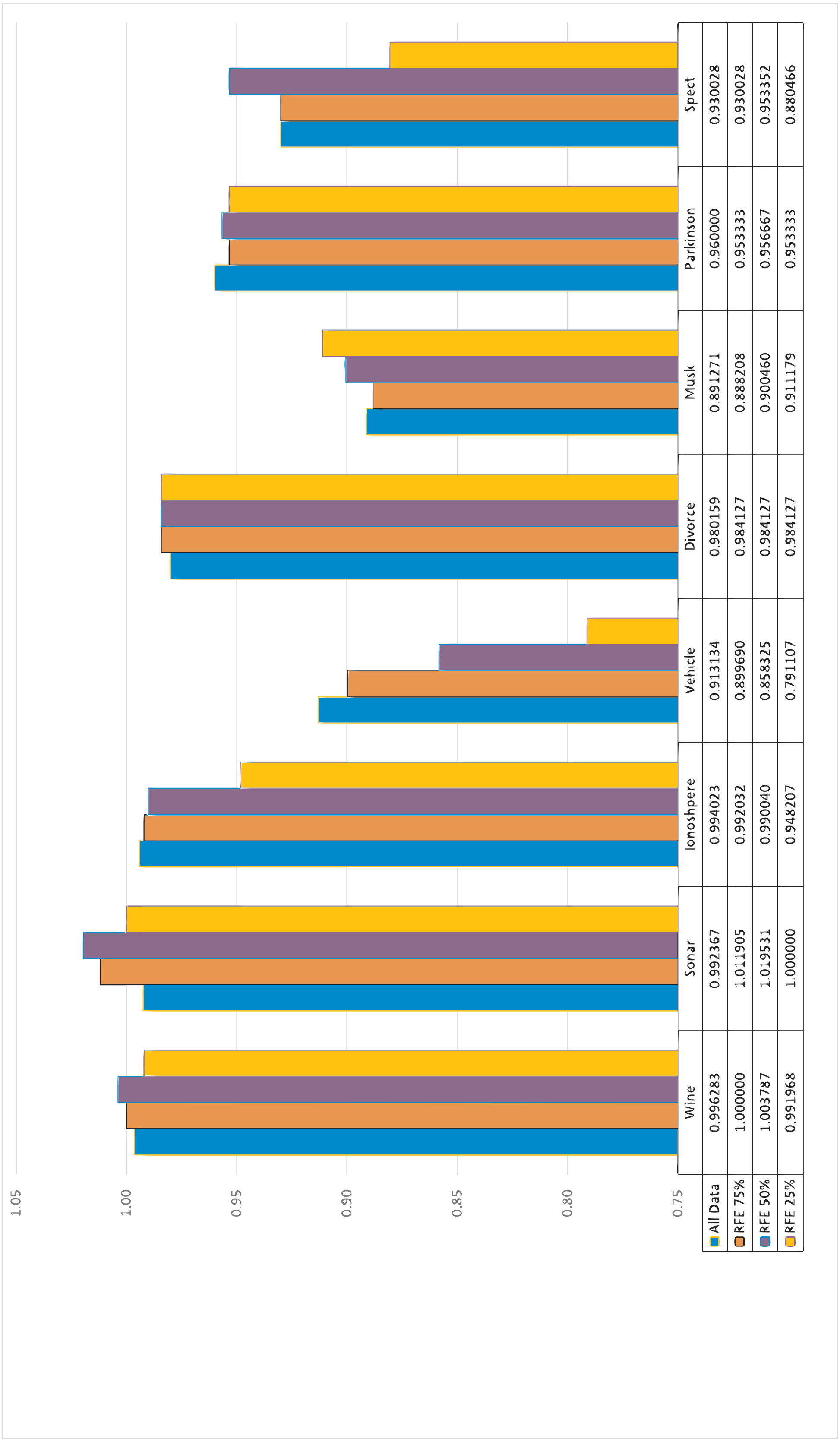
Acknowledgement

The data sets used in this article were obtained from UCI [34]. The citations [35] and [36] have been added upon data providers requests.



	Wine	Sonar	Ionoshpere	Vehicle	Divorce	Musk	Parkinson	Spect
🟦 All Data	0.999987	0.996183	0.992032	0.922441	0.984127	0.950995	0.953333	0.932944
🟧 RFE 75%	1.000000	1.003969	1.000000	0.936919	1.011905	0.960183	0.960395	0.932944
🟪 RFE 50%	0.996212	0.992188	1.000000	0.942090	1.011905	0.964778	0.946667	0.932944
🟨 RFE 25%	0.991968	0.995985	0.948207	0.786971	0.972222	0.961715	0.936668	0.927113

Fig. 2 Relative performance rates of RFE-HET-R-SVM



	Wine	Sonar	Ionoshpere	Vehicle	Divorce	Musk	Parkinson	Spect
All Data	0.996283	0.992367	0.994023	0.913134	0.980159	0.891271	0.960000	0.930028
RFE 75%	1.000000	1.011905	0.992032	0.899690	0.984127	0.888208	0.953333	0.930028
RFE 50%	1.003787	1.019531	0.990040	0.858325	0.984127	0.900460	0.956667	0.953352
RFE 25%	0.991968	1.000000	0.948207	0.791107	0.984127	0.911179	0.953333	0.880466

Fig. 3 Relative performance rates of RFE-HOM-R-SVM

Declarations

All authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding

No external funding was used in this study.

Conflicts of interest

The author declare that there are no conflicts of interest regarding the publication of this article.

Availability of data and material

No underlying data was collected or produced in this study.

Code availability

All simulation data and results are available upon request from any of the authors.

Author Contributions

All authors read and approved the final manuscript. Conceptualization: Uğur Madran, Duygu Soyoğlu; Methodology: Uğur Madran, Duygu Soyoğlu; Formal analysis and investigation: Uğur Madran, Duygu Soyoğlu; Writing - original draft preparation: Uğur Madran, Duygu Soyoğlu; Writing - review and editing: Uğur Madran, Duygu Soyoğlu; Funding acquisition: Uğur Madran, Duygu Soyoğlu; Resources: Uğur Madran, Duygu Soyoğlu; Software: Uğur Madran; Visualization: Duygu Soyoğlu.

References

[1] A. Ben-Dor, R. Shamir and Z. Yakhini, Clustering Gene Expression Patterns, *Journal of Computational Biology*, **6**, 3-4 (2004). doi: 10.1089/106652799318274

[2] R. Cooley, B. Mobasher and J. Srivastava, Data Preparation for Mining World Wide Web Browsing Patterns, *Knowledge and Information Systems*, **1**, 5-32 (1999). doi: 10.1007/BF03325089

[3] G. Punj and D.W. Stewart, Cluster Analysis in Marketing Research: Review and Suggestions for Application, *Journal of Marketing Research*, **20**, 134-148 (1983). doi: 10.2307/3151680

[4] O. Chapelle, B. Schölkopf and A. Zien, *Semi-Supervised Learning*, MIT Press: Cambridge, MA (2006).

[5] T. Yoshida, A graph-based approach for semisupervised clustering, *Computational Intelligence*, **30**, 263-284 (2014). doi: 10.1111/j.1467-8640.2012.00450.x

[6] N. Alon, M. Krivelevich, B. Sudakov, *Finding a Large Hidden Clique in a Random Graph* in SODA '98: Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, 594-598 (1998). doi: 10.5555/314613.315014

[7] G. Arslan, U. Madran and D. Soyoglu, An Algebraic Approach to Clustering and Classification with Support Vector Machines, *Mathematics*, **10**, Article no:1 (2022). doi: 10.3390/math10010128

[8] B.P.W. Ames, Guaranteed clustering and biclustering via semidefinite programming, *Math. Program.*, **147**, 429-465 (2014). doi: 10.1007/s10107-013-0729-x

[9] B.P.W. Ames and S.A. Vavasis, Convex optimization for the planted k-disjoint-clique problem, *Math. Program.*, **143**, 299-337 (2014). doi: 10.1007/s10107-013-0733-1

[10] B.P.W. Ames and S.A. Vavasis, Nuclear norm minimization for the planted clique and biclique problems, *Math. Program.*, **129**, 68-89 (2011). doi: 10.1007/s10107-011-0459-x

[11] R.D. Luce and A.D. Perry, A method of matrix analysis of group structure, *Psychometrika*, **14**, 95-116 (1949). doi: 10.1007/BF02289146

[12] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, **2**, 121-167 (1998). doi: 10.1023/A:1009715923555

[13] A.J. Smola and B. Schölkopf, A tutorial on support vector regression, *Statistics and Computing*, **14**, 199-222 (2004). doi: 10.1023/B:STCO.0000035301.49549.88

[14] V.N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed., Springer, Stanford, California (2008). doi: 10.1007/978-1-4757-3264-1

[15] V.N. Vapnik, *Statistical Learning Theory*, John Wiley and Sons, New York (1998).

[16] S.G. Chen and X.J. Wu, Multiple birth least squares support vector machine for multi-class classification, *Int. J. Mach. Learn. & Cyber.*, **8**, 1731-1742 (2017). doi: 10.1007/s13042-016-0554-7

[17] H. Chen, P.N. Tan and R. Jin, Efficient algorithm for localized support vector machine, *IEEE Transactions on Knowledge and Data Engineering*, **22**, 537-549 (2010). doi: 10.1109/TKDE.2009.116

[18] J. Wang, X. Wu and C. Zhang, Support vector machines based on K-means clustering for real-time business intelligence systems, *International Journal of Business Intelligence and Data Mining*, **1**, 54-64 (2005). doi: 10.1504/IJBIDM.2005.007318

[19] C. Zhao, C. Mi and Y. Shi, *Statistical Movement Classification based on Gram-Schmidt Transform* in 2021 International Conference on Electronic Information Technology and Smart Agriculture (ICEITSA), 142-147 (2021). doi: 10.1109/ICEITSA54226.2021.0003

[20] X. Zhu, Y. Wang, Y. Li, Y. Tan, G. Wang and Q. Song, A new unsupervised feature selection algorithm using similarity-based feature clustering, *Computational Intelligence*, **35**, 2-22 (2019). doi: 10.1111/coin.12192

[21] R. Sonoda, Fair oversampling technique using heterogeneous clusters, *Information Sciences*, **640**, (2023). doi: 10.1016/j.ins.2023.119059

[22] A. R. Pathare and A. S. Joshi, *Dimensionality Reduction of Multivariate Images Using the Linear & Nonlinear Approach* in 2023 International Conference on Device Intelligence, Computing and Communication Technologies, (DICCT), 234-237 (2023). doi: 10.1109/DICCT56244.2023.10110258
[23] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed., Springer: Stanford, California (2008). doi: 10.1007/978-0-387-84858-7
[24] V. Vapnik and A. Chervonenkis, *Theory of Pattern Recognition [in Russian]*, Nauka, Moscow (1974).
[25] Cheng H.; Tan, P.N.; Jin, R., *Localized Support Vector Machine and Its Efficient Algorithm,* in Proceedings of the 2007 SIAM International Conference on Data Mining (SDM), Radisson University Hotel Minneapolis, Minnesota, 26-28 April; Apte, C., Skillicorn, D., Liu, B., Parthasarathy S. doi: 10.1137/1.9781611972771.45
[26] M. Mesiter and I. Steinwart, Optimal learning rates for localized SVMs, *The Journal of Machine Learning Research*, **17**, 6722—6765 (2016).
[27] R. Rastogi, H. Safdari and S. Sharma, *Exploring Data Reduction Techniques for Time Efficient Support Vector Machine Classifiers* in 2018 IEEE Symposium Series on Computational Intelligence (SSCI), 2053–2059 (2018). doi: 10.1109/SSCI.2018.8628716
[28] Y. Chen, A. Jalali, S. Sanghavi and H. Xu, Clustering Partially Observed Graphs via Convex Optimization, *Journal of Machine Learning Research*, **15**, 2213-2238 (2014).
[29] O.N. Almasi and M. Rouhani, Fast and de-noise support vector machine training method based on fuzzy clustering method for large real-world datasets, *Turk. J. Elec. Comp.*, **24**, 219-233 (2016). doi: 10.3906/elk-1304-139
[30] D.R. Cutting, D.R. Karger, J.O. Pedersen, J.W. Tukey, *Scatter/Gather: a cluster-based approach to browsing large document collections* in SIGIR '92: Proceedings of the 15th annual international, 318-329 (1992). doi: 10.1145/133160.133214
[31] L. Van Der Maaten, E. Postma and J. Van den Herik, Dimensionality reduction: a comparative, *Tilburg University Technical Report*, TiCC-TR 2009-005 (2009).
[32] J.P. Cunningham and Z. Ghahramani, Linear dimensionality reduction: Survey, insights, and generalizations, *The Journal of Machine Learning Research*, **16**, 2859-2900 (2015).
[33] Pedregosa et al., Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research*, **12**, 2825-2830 (2011). https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html
[34] D. Dua and C. Graff, *UCI Machine Learning Repository*, University of California, Irvine, School of Information and Computer Sciences. (accessed on 18.09.2022). Available online: http://archive.ics.uci.edu/ml
[35] L. Naranjo, C.J. Pérez, Y. Campos-Roca and J. Martín, Addressing voice recording replications for Parkinson's disease detection, *Expert Systems With Applications*, **46**, 286-292 (2016). doi: 10.1016/j.eswa.2015.10.034
[36] M.K. Yöntem, K. Adem and T. İlhan, Divorce Prediction Using Correlation Based Feature Selection And Artificial Neural Networks, *Nevşehir Hacı Bektaş Veli Üniversitesi SBE Dergisi*, **9**, 259-273 (2019). https://dergipark.org.tr/en/pub/nevsosbilen/issue/46568/549416

Uğur Madran received the B.S., M.S., Ph.D. degrees in mathematics from Bilkent University, Ankara, Turkey, in 1998, 2000, and 2006 respectively. From 2006 to 2016, he was at Izmir University of Economics, Izmir, Turkey, and he has been an Associate Professor at the American University of the Middle East, Egaila, Kuwait since 2016. His original research interest was in commutative algebra, mainly in invariant theory, and recently, he is also interested in machine learning and data science. Mr. Ugur is a member of the American Mathematical Society, IEEE Computational Intelligence Society and the Turkish Mathematical Society, and he serves as the chair in the Department of Mathematics.

Duygu Soyoğlu was born in Izmir, Turkey in 1985. She received the B.S. and M.S. degrees in mathematics from Izmir University of Economics, Izmir, in 2010 and Yasar University, Izmir, in 2012, respectively, and the Ph.D. degree in Applied Mathematics and Statistics from Izmir University of Economics, Izmir in 2017. From 2016 to 2019, she was an instructor in Izmir University of Economics and since 2019, she is an Assistant Professor in American University of the Middle East, Egaila, Kuwait. Her research interest includes nonlinear partial differential and difference equations, mathematical physics and machine learning. Ms. Duygu is a member of the Association for Turkish Women in Mathematics, the American Mathematical Society and IEEE Computational Intelligence Society.