Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 115 (2017) 235–242

Procedia Computer Science

www.elsevier.com/locate/procedia

7th International Conference on Advances in Computing & Communications, ICACC 2017, 22-24 August 2017, Cochin, India

Diabetes-Finder: A Bat Optimized Classification System for Type-2 Diabetes

Damodar Reddy Edla[a], Ramalingaswamy Cheruku[a,*]

[a]*Department of Computer Science and Engineering, National Institute of Technology Goa, Ponda 403401, India*

Abstract

Type-2 Diabetes is one of the foremost causes for the increase in mortality across the world-wide. In this context, classification systems help doctors by analyzing the disease data. Radial Basis Function Neural Networks (RBFNN) are extensively used as classifier in medical domain because of its non-iterative nature. The size of the RBFNNs hidden-layer increases on par with dataset size. It's difficult to determining the optimal number of neurons in hidden-layer by cost effectively. In this paper, to address this problem, we have proposed Bat-based clustering algorithm. The proposed method experimented on Pima Indians Diabetes dataset and results outperform the competing approaches.

© 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the 7th International Conference on Advances in Computing & Communications.

Keywords: RBFNN; Type-2 Diabetes Classification; Bat Optimization; Class by Class Approach.

1. Introduction

Diabetes is a metabolic disease which causes due to deficiency of insulin hormone in human body. There are mainly two types of Diabetes namely type-1 and type-2. In type-1 Diabetes the body's immune system destroys the cells (β-cells) that release insulin, eventually it eliminates insulin production from the body. Insulin is a harmone that helps to cells in absorbing glucose, which they need to produce energy. Eventually body becomes insulin dependent. In type-2 diabetes, the body isn't able to use insulin in the right way. This situation is called insulin resistance. As type-2 Diabetes gets worse, the pancreas may produce less and less insulin. This is called insulin deficiency [1]. Type-2 Diabetes accounts for the vast majority share in Diabetes effected people (95 out of 100 people) [2].

Without insulin or insufficient insulin leads to excess sugar levels in the blood. As a result, it impacts lot of organs in the body such as heart, kidney, limp, eye etc. In the long run this situation leads to a various complications like cardiovascular damage, kidney damage, nerve damage, eye damage and stroke etc. [3, 4]. Hence, it is necessary to use more robust classifier which helps the doctors for better diagnosis of diabetes in early stages.

* Corresponding author. Tel.: +91-957-382-7143.
E-mail address: dr.reddy@nitgoa.ac.in, rmlswamygoud@gmail.com

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the 7th International Conference on Advances in Computing & Communications
10.1016/j.procs.2017.09.130

In the literature, many people already used classification and predictive systems in the health care sector to explore hidden patterns in the patients data. These systems aid, medical professionals to enhance their diagnosis procedures. Though iterative models like Multi-Layer Feed Forward Neural Network (MLFFNN), Multi-Layer Perceptron Network (MLPN) etc. are most popular in machine learning they requires more time for network convergence. Unlike MLFFNN, MLPN the RBFNNs are non-iterative models requires single iteration for training. Due to non-iterative nature of RBFNNs became popular for classification task. And, the performance of these RBFNNs are on par with the more widely used MLFFNN and MLPN models [5]. These RBFNNs are made up of three layers namely input layer, hidden layer and output layer.

The size of the input layer is determined by the dimensionality of training patterns and output layer is by number of distinct classes in training patterns. To figure out number of neurons in the hidden layer, the simplest and most common method is to assign a neuron for each training pattern. Though this process is simple, it is not practical since most of the real time applications have numerous training patterns with high dimensionality. So, usually it is a good practice to cluster the training patterns first to create a reasonable number of groups by employing clustering techniques such as *k*-means, *k*-medoids, SOM, etc. Once we create a group we can assign a neuron to each group (cluster).

In the process of identifying the number of clusters in a given data, there is a problem of more computational cost. Moreover, there is a problem of having different class data under same cluster. Especially at the cluster center locations near line of separation. Applying clustering technique on whole datast requires more computation power and causes degradation in RBFNN performance due to unbalanced clustering near line of separation. Hence, to address these problems in this paper we have proposed Bat-based clustring. The proposed Bat-based clustering is applied in class by class fashion instead of whole dataset. This approach not only reduces the computation time (due to less number of patterns) but also increases RBFNN performance (by avoiding un balanced clustering) [6].

As a way to identify size of RBFNN hidden layer, we need to fix a number of neurons from each class along with their radial basis function characteristics. Normally, these radial basis functions are Gaussian functions. A Gaussian function is usually characterized by its center location and spread. To find these center locations for Gaussian functions earlier so many attempts made by using clustering techniques such as *k*-means, *k*-medoids, SOM etc. Once membership of all data points are determined using Bat-based clustering approach, for each cluster average of cluster elements is treated as the center and variance of cluster elements is treated as spread. These centers and variances values obtained using Bat-based clustering approach are used as inputs for basis (Gaussian) functions [6].

The rest of the paper is organized as follows: related work is discussed in Section 2. Th proposed method is discussed in Section 3. The experimental results are shown in Section 4. The Section 5, concludes the paper based on experimental observations.

2. Related work

As a way to reduce the complexity of RBFNN, J. Petez [7] has proposed Dynamic Decay Adjustment model i.e. RBFN-DDA. This model reduced the complexity of network by up to 93.9% (in terms of number of neurons). However, the drawback of RBFN-DDA is its greedy insertion behavior. In 2013, Qasem et al. [8] have proposed an algorithm named as memetic Multiobjective Particle Swarm Optimization RBF Network (MPSON). It integrates the accuracy and structure of an RBFNN. The MPSON generated RBFNNs coming with an appropriate balance between accuracy and simplicity. F. Cruz et al. [9] have used Artificial Bee Colony (ABC) inspired clustering approach to design RBFNN. They have used cOptBees, plus a heuristic to automatically select the number,location and dispersions of basis functions to be used in RBFNN. They have experimented on eight classification datasets. Cheruku et al. [6] integrated Cluster Validity Index (CVI) with *k*-means algorithm to generate class wise optimal clusters. This model is experimented on PID dataset and synthetic datasets. Their model outperformed in terms of accuracy, classification time and training times. Moreover, it reduced the complexity of RBFNN nearly 94%.

3. Proposed methodology

3.1. Proposed Radial Basis Function Neural Network Architecture

The proposed RBFNN [6] is a three layer feed forward architecture as shown in Fig. 1. Construction of this RBFNN involves determination of number of neurons in each layer.

- **Input layer:** The input layer is made with D number of neurons, where D is the dimensionality of patterns. This layer is fully connected to hidden layer in feed forward fashion. And there are no weights associated with these inter connected links so there is no transformation happen at this layer. It simply forward the inputs to hidden layer.
- **Hidden layer:** The hidden layer is made up of $H=P+Q$ number of neurons, where P is the negative class centers and Q is the positive class centers. These hidden layer neurons are complexly connected with 2 output layer neurons. At each hidden layer neuron, it is a nonlinear transformation because of Gaussian activation function. The output value of each hidden layer neuron in negative class and positive class are computed using Eq. (1) and Eq. (2) for input pattern I respectively.

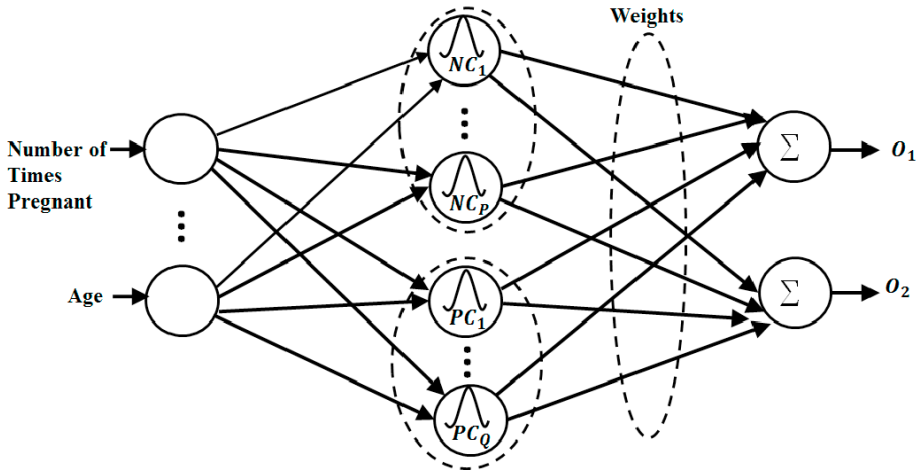

Fig. 1. The RBFNN for Diabetes classification.

$$\varphi_i^-(I) = \frac{1}{\sqrt{2\Pi}\sigma_i^-} e^{-\frac{\left|I-\mu_i^-\right|}{2\left(\sigma_i^-\right)^2}}, i = 1, 2, \ldots, P. \tag{1}$$

$$\varphi_i^+(I) = \frac{1}{\sqrt{2\Pi}\sigma_i^+} e^{-\frac{\left|I-\mu_i^+\right|}{2\left(\sigma_i^+\right)^2}}, i = 1, 2, \ldots, Q. \tag{2}$$

Where $\left|I-\mu_i^-\right|$ and $\left|I-\mu_i^+\right|$ are the euclidean distances from pattern to negative and positive class cluster center respectively.

- **Output layer:** The output layer is made up with *2* number of neurons (number of neurons are determined by number of classes in the dataset). Transformation at this layer is linear because response of the output layer neuron is a weighted sum of hidden layer outputs, which is computed using Eq. (3).

$$0_j(I) = \sum_{i=1}^{P} w_{ji}\phi_i^-(I) + \sum_{i=1}^{Q} w_{ji}\phi_i^+(I), \mathrm{j} = 1, 2 \tag{3}$$

Once we obtain the output values of output layer neurons, class label is assigned based on max operator. It is shown in Eq. (4) for input pattern I.

$$\text{class-label(I)} = arg\max_j O_j(I), \mathrm{j}=1, 2 \tag{4}$$

The weight vector $w = [w_{11}, w_{12}, \ldots, w_{1P}, w_{21}, w_{22}, \ldots, w_{2Q}]^T_{1\times H}$ between ouput layer and hidden layer is given by

$$w = \phi^+_{L\times H} * T \tag{5}$$

Where ϕ^+ is the pseudoinverse of the ϕ matrix

$$\phi = \begin{bmatrix} \phi_1^-(I_1) & \ldots & \phi_P^-(I_1) & \phi_1^+(I_1) & \ldots & \phi_Q^+(I_1) \\ \phi_1^-(I_2) & \ldots & \phi_P^-(I_2) & \phi_1^+(I_2) & \ldots & \phi_Q^+(I_2) \\ & & & \cdot & & \\ \phi_1^(I_L) - & \ldots & \phi_P^-(I_L) & \phi_1^+(I_L) & \ldots & \phi_Q^+(I_L) \end{bmatrix}_{L\times H} \tag{6}$$

Where L is number of training patterns and H is number of neurons in hidden layer.

3.2. Bat optimization Algorithm (BA) preliminaries

BA [10, 6] simulates the behavior of microbats. These microbats uses a type of sonar called echolocation to detect the prey, and avoid obstacles in the dark. These bats emit a very loud sound pulse and listen for the echo that bounces back from the surrounding objects. In BA, each solution (cluster centers) is represented by bat. Every bat has fitness value which is evaluated by using fitness function. The fitness function used in our experiments is given in Eq. (8). The aim of BA is to maximize the fitness value in order to find optimal solution (cluster centers)in the search space.

$$\text{fitness function} = arg \min_K \text{Intra-Inter Validity} \tag{7}$$

The fitness function proposed in Eq. (8) is adopted from Ray and Turi [11, 12] validity index. It uses both intra-cluster and inter-cluster distances in order to search for optimal clusters inside the data. According to Ray and Turi the Intra-Inter validity index defined as follows:

$$\text{Intra-Inter Validity} = \frac{\text{Intra cluster distance}}{\text{Inter cluster distance}} \tag{8}$$

Where,
$Intraclusterdistance = \frac{1}{N}\sum_{i=1}^{k}\sum_{x\in c_i} ||x - z_i||^2$ and
$Interclusterdistance = \min_{i,j}\left(\left|z_i - z_j\right|^2\right)$

Where, $i = 1, 2, \ldots, k-1$, $j = i+1, i+2, \ldots,$ k, z_i denotes the center of the cluster c_i, k is the number of the clusters and N is the number of data points.

Intra-Inter validity index needs to be minimized for better clusters. It means that smaller value for index indicates, intra comapct clusters and well separated inter clusters.

The basic steps of the BA based clustering is summarized in Algorithm 1 as a pseudo code.

3.3. BA-based clustering

The pseudo code for BA-based clustering algorithm shown in Algorithm 1. This Algorithm 1 takes maximum number of clusters (*NC* for negative class and *PC* for positive class) and training dataset as input and outputs optimal clusters center locations.

4. Experimental results and discussion

All the experiments are conducted using Matlab R2015a tool on system with RAM size of 8 Gb and Intel i7 processor speed of 3.6 GHz. We have used measures like classification accuracy, sensitivity, specificity, network complexity (in terms of number of connections), computational time (for finding the clusters center), and classification time to evaluate the performance of the proposed model. In our experiments we have considered $NC = PC = 50$.

Algorithm 1: Pseudo code for BA-based clustering.

Input: Class specific training dataset, maximum number of clusters;
Output: Best clusters center for given class;

Fitness function $f(B_{1,2,...,D})$, where D is the dimensionality ;
Initialize parameter values A_{min}, A_{max}, r, F_{min}, F_{max}, B_{max}, I_{max}, γ, α from Table 3;
$K \leftarrow$ NC for negative class (or) PC for positive class ;
for $k \leftarrow 2$ ***to*** K **do**
    $FitnessArray \leftarrow [\ ]$;
    Initialize the bat population ;
    for $i \leftarrow 1$ ***to*** B_{max} **do**
        Initialize each Bat ;
        $F_i = F_{min}$ + rand(k) * [F_{max}-F_{min}];
        $V_i^t = V_i^{(t-1)} + (B_i^t - B_*)\ Q_i$;
        $B_i^t = B_i^{(t-1)} + V_i^t$;;
        Compute the fitness of each bat;
        FitnessArray= FitnessArray $\cup$ f(B_i^t)
    Find the BestBat (B_*)= min(FitnessArray) ;
    Iter =1;
    while $Iter \leq I_{max}$ **do**
        for $i = 1$ *to* B_{max} **do**
            Generate new solutions by adjusting frequency and updating velocities and locations/solutions;
            if $(rand \geq r_i)$ **then**
                Compute $A^t\epsilon$, Where $\epsilon \in [1, 1]$
                And calculate local solution using below equation
                $B_{new} = B_{old} + A^t\epsilon$;
            Generate a new solution by flying randomly;
            if $(A_i \geq rand)$ *and* $(f(B_*) \geq f(B_i))$ **then**
                Calculate the new solution by
                Increasing r_i using $r_i^{(t+1)} = r_i^o\ [1 - exp^{(-\gamma t)}]$;
                and Decreasing A_i by α using $A_i^{(t+1)} = \alpha A_i^t$;
        Find best bat solution corresponding to B_* for current iteration ;
    Find the best bat (B_*) corresponding to k.;
Find the global best bat (B_*) all over K;
return (B_*);

Table 1. Pima Indians Diabetes data set attributes description.

Feature S. No	Feature Description
F_1	Number of times pregnant
F_2	Plasma glucose concentration
F_3	Diastolic blood pressure
F_4	Triceps skin fold thickness
F_5	Serum insulin
F_6	Body mass index
F_7	Diabetes pedigree function
F_8	Age
Class label	0 if Type-2 Diabetes is negative
	1 if Type-2 Diabetes is positive

We have used Pima Indians Diabetes (PID) dataset obtained from UCI machine learning repository [13] whose detail specifications are shown in Table 1. There are 8 conditional attributes, 1 decision attribute. PID dataset consisting of a total of 768 diabetes patients data in which 500 records are related to diabetes negative (class label 0) and 268 records are related to diabetes positive (class label 1). For experimental purpose the PID dataset partitioned into training and testing datasets. The training dataset is constituted with 538 patterns (350 class 0 patterns and 188 class 1 patterns) and testing dataset is constituted with remaining patterns (150 class 0 patterns and 80 class 1 patterns). In order to evaluate the performance of the proposed method accuracy, sensitivity and specificity measures are defined in Eqs. (9)-(11) respectively. The confusion matrix parameters used in these equations are defined in Table 2.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{9}$$

$$Sensitivity = \frac{(TP)}{(TP + FN)} \tag{10}$$

$$Specificity = \frac{(TN)}{(TN + FP)} \tag{11}$$

Table 2. Description of confusion matrix parameters.

Parameter	Description
True Positive (TP)	Number of patterns classified as diabetes negative which are actually diabetes negative
True Negative (TN)	Number of patterns classified as diabetes positive which are actually diabetes positive
False Negative (FN)	Number of patterns classified as diabetes positive which are actually diabetes negative
False Positive (FP)	Number of patterns classified as diabetes negative which are actually diabetes positive

In order to obtain the best performance from BA, It is necessary to fine tune BA parameters. These fine tuned parameters are shown in Table 3.

Table 3. Fine tuned parameters for Bat optimization algorithm.

S.No	Parameter	Value	Explanation
1	B_{max}	50	No. of Bats
2	A	[0 , 2]	Sound Loudness
3	$[F_{min}, F_{max}]$	[0, 10]	Frequency range
4	I_{max}	1000	Maximum Iterations
5	r	0.1	Pulse Rate
6	γ	0.9	Increment Value of Pulse Rate
7	α	0.9	Decrement Value of Loudness

The Bat-based clustering in applied on each class training dataset to generate best locations for cluster centers. These results are furnished in Table 4.

Table 4. The proposed model performance on PID dataset

Dataset	Number of clusters		
	-ve class	+ ve class	Total
PID	31	16	47

Once we obtain the class wise optimal cluster locations according to Algorithm 1, the proposed model is constructed and experimented on PID dataset. The performance of PID dataset on test dataset is shown in Table 5.

The proposed model is compared with MLFFNN, CFN (Cascade Forward Networks), TDN (Time Dealy Network), Leraning Vector Quantization (LVQ), GINI, AIS (Artificial Immune System) and Probabilistic Neural Network (PNN)

Table 5. The proposed model performance on PID dataset

		Predicted class		
		+ve	- ve	Total
Actual class	+ve	TP = 122	FP = 28	150
	-ve	FN = 32	TN = 48	80
	Total	151	79	230

Table 6. Comparison of proposed method with other classification algorithms.

Model	Accuracy (%)	Sensitivity (%)	Specificity (%)	Reference
GINI	65.97	44.71	77.78	M. R. Bozukurt et al. [14]
TDN	66.80	41.11	81.25	M. R. Bozukurt et al. [14]
CFN	68.00	62.22	71.25	M. R. Bozukurt et al. [14]
MLFFNN	68.80	54.44	76.88	M. R. Bozukurt et al. [14]
LVQ	73.60	54.44	84.38	M. R. Bozukurt et al. [14]
AIS	68.80	52.22	78.13	M. R. Bozukurt et al. [14]
PNN	72.00	63.33	76.88	M. R. Bozukurt et al. [14]
Proposed model	**73.91**	**81.33**	60.00	This study

Table 7. Performance comparison between proposed RBFNN and conventional RBFNN.

	Conventional RBFN	Diabetes-Finder
Hidden layer size	768	47
Network complexity	7680	470
% Reduction in network complexity	0%	93.88%
Accuracy (%)	68.53	**73.91**
Classification time (seconds)	0.46	0.05
Computational time (seconds)	269.75	56.20

in terms of accuracy, sensitivity and specificity. These results are shown in Table 6. It is observed from the table results that proposed RBFNN model outperformed in terms of accuracy, sensitivity ans specificity over all these classifiers. Further, the proposed model is compared with conventional RBFNN in terms of hidden layer size, network complexity and so on. These comparison results are shown in Table 7. From the table results it is clear that proposed model reduced the network complexity nearly 94%. Moreover, it also reduced the hidden layer size, classification time and computational time. However, it increased the RBFNN performance compared to conventional RBFNN.

Finally, to validate the robustness of proposed model, it is compared with other variants of RBFNN from similar problem domain such as Bee-RBF, MEPGAN-f1f2, CVI-RBFNN etc. These models are compared in terms of accuracy and hidden layer size (number of cluster centers). These results are provided in Table 8. It clear from the results that proposed model obtained better accuracy with few hidden layer neurons.

Table 8. Comparison of proposed method with other RBFNN variants of same domain.

Model	Accuracy (%)	# centers	Ref. (year)
RBFN-DDA-T	73.50	65	J. Paetz (2004) [7]
MEPGANf1f2	72.78	54	S. N. Qasem et al. (2013) [8]
Bee-RBF	71.13 ±1.06	35	F. Cruz et al. (2016) [9]
Dunn Index-RBFNN	69.13	48	Cheruku et al. (2017) [6]
Proposed model	73.91	47	This study

5. Conclusion

In this paper, we have proposed Diabetes-Finder for effective diagnosis of Type-2 Diabetes. The proposed model constructed the RBFNN classification system by obtaining the class wise cluster centers i.e., class by class approach. The proposed model is experimented on PID dataset. Due to class by class approach the computational time is reduced (due to less number of patterns) and accuracy is increased (by avoiding un balanced clustering). Experimental results on PID data set proved that proposed model with few neurons in hidden layer outperformed in terms of accuracy compared to PNN, MLFFNN, CFN, TDN, GINI, AIS, LVQ and conventional RBFNN. Proposed model reduced not only the complexity of network by 93.88%, but also training time to below two seconds and classification time to 0.05 seconds.

Further, the proposed method achieved best performance when compared to other variants of RBFNN. Overall, the proposed RBFNN increased the accuracy, reduced the computational time and testing time drastically.

Finally, by observing all the results we conclude that proposed Diabetes-Finder model achieved best accuracy, by reducing the network complexity, computational time and classification time drastically. This helps to create simple and powerful RBFNN for the purpose of classification task.

References

[1] Diabetes overview: Differences between type-1 and type-2:. `http://www.webmd.com/diabetes/tc/diabetes-differences-between-type-1-and-2-topic-overview`. Accessed: 2016-09-30.

[2] WHO. World health organization:. `http://www.who.int/diabetes/action_online/basics/en/`. Accessed: 2016-09-30.

[3] Ramalingaswamy Cheruku, Damodar Reddy Edla, and Venkatanareshbabu Kuppili. Sm-ruleminer: Spider monkey based rule miner using novel fitness function for diabetes classification. *Computers in Biology and Medicine*, 81:79–92, 2017.

[4] Ramalingaswamy Cheruku, Damodar Reddy Edla, Venkatanareshbabu Kuppili, and Ramesh Dharavath. Rst-batminer: A fuzzy rule miner integrating rough set feature selection and bat optimization for detection of diabetes disease. *Applied Soft Computing*, 2017.

[5] B Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[6] Ramalingaswamy Cheruku, Damodar Reddy Edla, and Venkatanareshbabu Kuppili. Diabetes classification using radial basis function network by combining cluster validity index and bat optimization with novel fitness function. *International Journal of Computational Intelligence Systems*, 10(1):241–265, 2017.

[7] Jürgen Paetz. Reducing the number of neurons in radial basis function networks with dynamic decay adjustment. *Neurocomputing*, 62:79–91, 2004.

[8] Sultan Noman Qasem, Siti Mariyam Shamsuddin, Siti Zaiton Mohd Hashim, Maslina Darus, and Eiman Al-Shammari. Memetic multiobjective particle swarm optimization-based radial basis function network for classification problems. *Information Sciences*, 239:165–190, 2013.

[9] Dávila Patrícia Ferreira Cruz, Renato Dourado Maia, Leandro Augusto da Silva, and Leandro Nunes de Castro. Beerbf: a bee-inspired data clustering approach to design rbf neural network classifiers. *Neurocomputing*, 172:427–437, 2016.

[10] Xin-She Yang. A new metaheuristic bat-inspired algorithm. *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pages 65–74, 2010.

[11] Siddheswar Ray and Rose H Turi. Determination of number of clusters in k-means clustering and application in colour image segmentation. In *Proceedings of the 4th international conference on advances in pattern recognition and digital techniques*, pages 137–143. Citeseer, 1999.

[12] Judong Shen, Shing I Chang, E Stanley Lee, Youping Deng, and Susan J Brown. Determination of cluster number in clustering microarray data. *Applied Mathematics and Computation*, 169(2):1172–1185, 2005.

[13] M. Lichman. UCI machine learning repository, 2013.

[14] Mehmet Recep Bozkurt, NİLÜFER YURTAY, Ziynet Yilmaz, and Cengiz Sertkaya. Comparison of different methods for determining diabetes. *Turkish Journal of Electrical Engineering & Computer Sciences*, 22(4):1044–1055, 2014.