Turkish Journal of Electrical Engineering and Computer Sciences

Volume 22 | Number 4 Article 17

1-1-2014

Comparison of different methods for determining diabetes

MEHMET RECEP BOZKURT

NİLÜFER YURTAY

ZİYNET YILMAZ

CENGİZ SERTKAYA

Turkish Journal of Electrical Engineering & Computer Sciences

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

Research Article

Turk J Elec Eng & Comp Sci
(2014) 22: 1044 – 1055

doi:10.3906/elk-1209-82

Comparison of different methods for determining diabetes

Mehmet Recep BOZKURT[1,*], **Nilüfer YURTAY**[2], **Ziynet YILMAZ**[1], **Cengiz SERTKAYA**[2]

[1]Department of Electric & Electronics Engineering, Sakarya University, Sakarya, Turkey
[2]Department of Computer Engineering, Sakarya University, Sakarya, Turkey

Received: 18.09.2012 • **Accepted:** 01.01.2013 • **Published Online:** 17.06.2014 • **Printed:** 16.07.2014

Abstract: In this study, the Pima Indian Diabetes dataset was categorized with 8 different classifiers. The data were taken from the University of California Irvine Machine Learning Repository's web site. As a classifier, 6 different neural networks [probabilistic neural network (PNN), learning vector quantization, feedforward networks, cascade-forward networks, distributed time delay networks (DTDN), and time delay networks], the artificial immune system, and the Gini algorithm from decision trees were used. The classifier's performance ratios were studied separately as accuracy, sensitivity, and specificity and the success rates of all of the classifiers are presented. Among these 8 classifiers, the best accuracy and specificity values were achieved with the DTDN and the best sensitivity value was achieved with the PNN.

Key words: Diabetes diagnosis, artificial neural networks, decision tree, artificial immune system, classification, Pima Indian

1. Introduction

Studies in the field of medical decision support systems have been established and, due to the high success rate of these studies, interest in this field is increasing every day. These systems frequently use various artificial intelligence techniques and data mining.

In medical decision support systems, there is a greater interest in the study of diseases that are common throughout the world, and diabetes is one of them. Previous studies in this field are mentioned below.

Masharani and Karam stated that diabetes mellitus is a syndrome with disordered metabolism and inappropriate hyperglycemia due to either a deficiency of insulin secretion or the combination of insulin resistance and inadequate insulin secretion to compensate [1].

There are 2 main types of diabetes: type 1 and type 2. Diabetes is one of the most common chronic diseases. According to the current data from the World Health Organization, 346 million people worldwide have diabetes and 80% of those people live in low- or middle-income countries [2].

A study by Huang et al., based on data collected between 2000 and 2004 from Ulster Community and Hospitals Trust (UCHT), aimed to the detect key determinants of type-2 diabetes using data mining techniques [3].

In another study, Lekkas et al. addressed the diagnosis of Pima Indian diabetes using the evolving fuzzy approach. An accuracy of 79.35% was achieved for this diagnosis [4].

In the literature, some other studies for the classification of data regarding Pima Indian diabetes were

*Correspondence: mbozkurt@sakarya.edu.tr

made. Polat et al. using a PCA-adaptive neuro-fuzzy inference system [5], Temurtas et al. using a multilayer neural network (MLNN) [6], Kayaer and Yildirim using a general regression neural network (GRNN) [7], and Meng et al. using the artificial immune recognition system (AIRS) [8] performed classification on these data and achieved various percentage values. However, none of the above studies, except [5], mentioned sensitivity and specificity values. Only the accuracy values were given.

Since Pima Indians are the most intense population with type-2 diabetes in the world, data from this population is widely used in diabetic studies. This study and some of the studies mentioned above also used Pima Indian diabetes data from the University of California Irvine (UCI) Machine Learning Repository's web site.

Classification was done on these data using the artificial immune system (AIS), Gini algorithm from decision trees, and 6 different artificial neural network (ANN) algorithms, which are as follows: distributed time delay networks (DTDNs), time delay networks (TDNs), probabilistic neural networks (PNNs), cascade-forward networks (CFNs), feedforward networks (FFNs), and learning vector quantization (LVQ).

The performances of these 8 classifications were compared with each other. In addition to the implementation of different methods in this study, receiver operating characteristic (ROC) analysis was included, even though it was not included in other studies.

2. Materials and methods

2.1. Dataset

The Pima Indians, Native Americans who live around Arizona, are the most intense type-2 diabetic population in the world. Since it is a homogeneous group, the data taken from these people are the subject of intense studies in diabetics [9].

Table 1 shows the properties of the data. Accordingly, 9 attributes (8 input and 1 output) were studied. Output information or class values are indicated as 0: no diabetes and 1: diabetes [10].

Table 1. Pima Indian Diabetes dataset attributes.

Attribute no.	Attribute
1	Number of times pregnant
2	Plasma glucose concentration
3	Diastolic blood pressure (mmHg)
4	Triceps skin-fold thickness (mm)
5	2-h serum insulin (mu U/mL)
6	Body mass index (kg/m^2)
7	Diabetes pedigree function
8	Age
9	Class 0 or 1

2.2. AIS

The AIS emerged in the 1990s as a biologically based calculation method [11]. The system's characteristics of the learning algorithm were inspired from the ability to recognize and destroy germs in the human body [12]. Learning is implemented with the activity of lymphocytes, natural antibody production, preexisting immunity, wallet selection, tolerance, and memory [13].

In our study, the AIS algorithm was implemented in C# and compiled using VisualStudio.Net 2010.

The framework of our algorithm design is shown in Figure 1, which starts with creating the permanent population of antibodies using the input data.

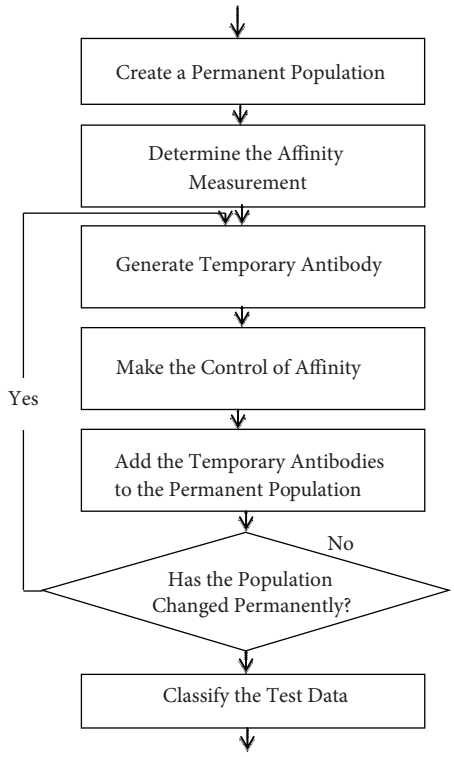

Figure 1. System diagram of positive cloning algorithm.

The AIS algorithm's success depends on choosing the right parameter values used during the implementation of the steps. The parameters used in the AIS are listed below.

Affinity threshold value: The measurement for controlling the healthy production of the temporary antibodies that were cloned. The affinity is chosen by selecting a value between 0 and 1. This is a numerical value that was chosen as 0.5 for this study.

Rate of cloning: Refers to the maximum number of clones that can be derived through an antibody in the cloning process. This is a numerical value that was chosen as 2 for this study.

Mutation rate: Represents the number of properties that will be mutated during the cloning process. This numerical value was determined as 2 for this study.

Threshold value: The accepted similarity value that is used for the comparison of the test data, which is desired to be classified with the antibody cells. This is a numerical value that was determined as 0.5 for this study.

The AIS algorithm parameters that were used are shown in Table 2.

Table 2. Parameters used in the AIS algorithm.

Parameters	Parameter value
Affinity threshold value	0.5
Rate of cloning	2
Mutation rate	2
Threshold value	0.5

According to the complexity of the problem, these parameters can be changed. As a result of various experiments, the best results were obtained from these parameter values. Therefore, the values in Table 2 were used in this model.

By mutating the antibody population that was generated, temporary antibodies were created. Two different antibodies of the same class were selected for the cloning process. In the process of mutation, a new antibody is created by changing the characteristic of the first antibody to the same characteristic of the second antibody. This process is shown in Table 3 with samples from the AIS input dataset. The newly created antibody is called a temporary antibody. The explained logic is also used in genetic algorithms. Two different antibodies of the same class are selected for the cloning process.

Table 3. Production of the temporary antibodies.

Features	→	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
First antibody	→	6	148 ↓	72	35	0	33.6 ↓	0.627	50
Second antibody	→	1	85	66	29	0	26.6	0.351	31
Temporary antibody	→	1	148	66	29	0	33.6	0.351	31

The temporary antibodies produced were tested with the affinity measurement. Accordingly, the similarities of the temporary antibody population with the permanent antibody population were calculated for each antibody. This was done using the Euclidean distance, as shown in Eq. (1) [14]:

$$\text{Euclid}(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}, \tag{1}$$

where n is the number of features, x is the permanent antibody vector, and y is the temporary antibody vector.

With the similarity values obtained, Eq. (2) was used to determine the health condition of the antibody.

$$\text{Affinity size} = 1 - \text{Euclid}(x,y) = 1 - \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2}$$

If the most closely resembled and the produced antibody class of the temporary antibody are the same, and the affinity measure is above the threshold value, then it is accepted and included in the permanent population of antibodies. In other cases, the temporary antibody is destroyed.

The permanent antibody production process is continued until the population is steady. When the permanent antibody population is steady, the testing values are ready for classification.

In this case, the test data and the permanent antibody population are compared using Eq. (1).

Among the antibodies whose likelihood ratios are over the threshold, the antibody class with the highest similarity ratio is determined as the class of the test data that is needed to be classified.

Principal component analysis (PCA) − based k-nearest neighbor (k-NN) analysis was used in this study. As a result of this classification, the following was obtained: sensitivity 52.22%, accuracy 68.8%, and specificity 78.13%.

2.3. Decision tree

The increase of data held in electronic environments has caused the emergence of various questions, such as how and where to use the data, how to interpret it, and how to access information. Data mining is a study area

to find the answers to these questions, including techniques such as classification, clustering, and association analysis. There are many methods used in the classification technique. Among these methods, the decision tree is widely used. It is possible to create many decision trees with algorithms such as ID3, C4.5, Gini, and Twing [15].

The Gini algorithm is based on the separation of the attribute values to the left and right of the form of binary divisions. The obtained values are grouped into classes corresponding to the left and right divisions. Each node in the calculations is carried out separately for the left and right divisions (Gini_{left}, Gini_{right}).

$\mathbf{L}_i$: On the left branch, i, the sample group(s) number

$\mathbf{R}_i$: i group in the right branch sample(s) number

k: The number of classes

T: Node samples

$|\mathbf{T}_{left}|$: Left branch sample(s) number

$|\mathbf{T}_{right}|$: Right branch sample(s) number

To be calculated with the following definitions of relations:

$$Gini_{left} = 1 - \sum_{i=1}^{k} \left(\frac{L_i}{|T_{left}|} \right)^2 \qquad Gini_{right} = 1 - \sum_{i=1}^{k} \left(\frac{R_i}{|T_{right}|} \right)^2. \tag{3}$$

The nature of the learning set for each j, the number of elements to be calculated, uses the following correlation:

$$Gini_j = \frac{1}{n} \left(|T_{left}| \times Gini_{left} + |T_{right}| \times Gini_{right} \right). \tag{4}$$

In this study, we created a decision tree based on the Gini algorithm using the Rapidminer program [16].

In Figure 2, a decision tree is given, with which the rules were obtained. These rules are the nature of the decision. The decision tree was created by using the Gini algorithm.

Expansion of abbreviations in Figure 2 is as follows:

a- Number of times pregnant

b- Plasma glucose concentration

c- Diastolic blood pressure (mmHg)

d- Triceps skin fold thickness (mm)

e- 2-h serum insulin (μU/mL)

f- Body mass index (weight in kg / (height in m)$\wedge$2)

g- Diabetes pedigree function

h- Age (years)

2.4. ANNs

ANNs have been improved with the objective of carrying out automatic computer operations, such as the derivation of new information, similar to the learning features of the brain. However, this task is considerably difficult with traditional programming methods. Thus, it is reasonable to say that ANNs are a type of adaptive information processing in computer science that was improved for programming very difficult events [17].

An ANN consists of training and testing phases. The network learns the relationships among the data being used in the training phase. In the testing phase, there are no conclusion data. The network gives the decision using the learned information on the training phase.

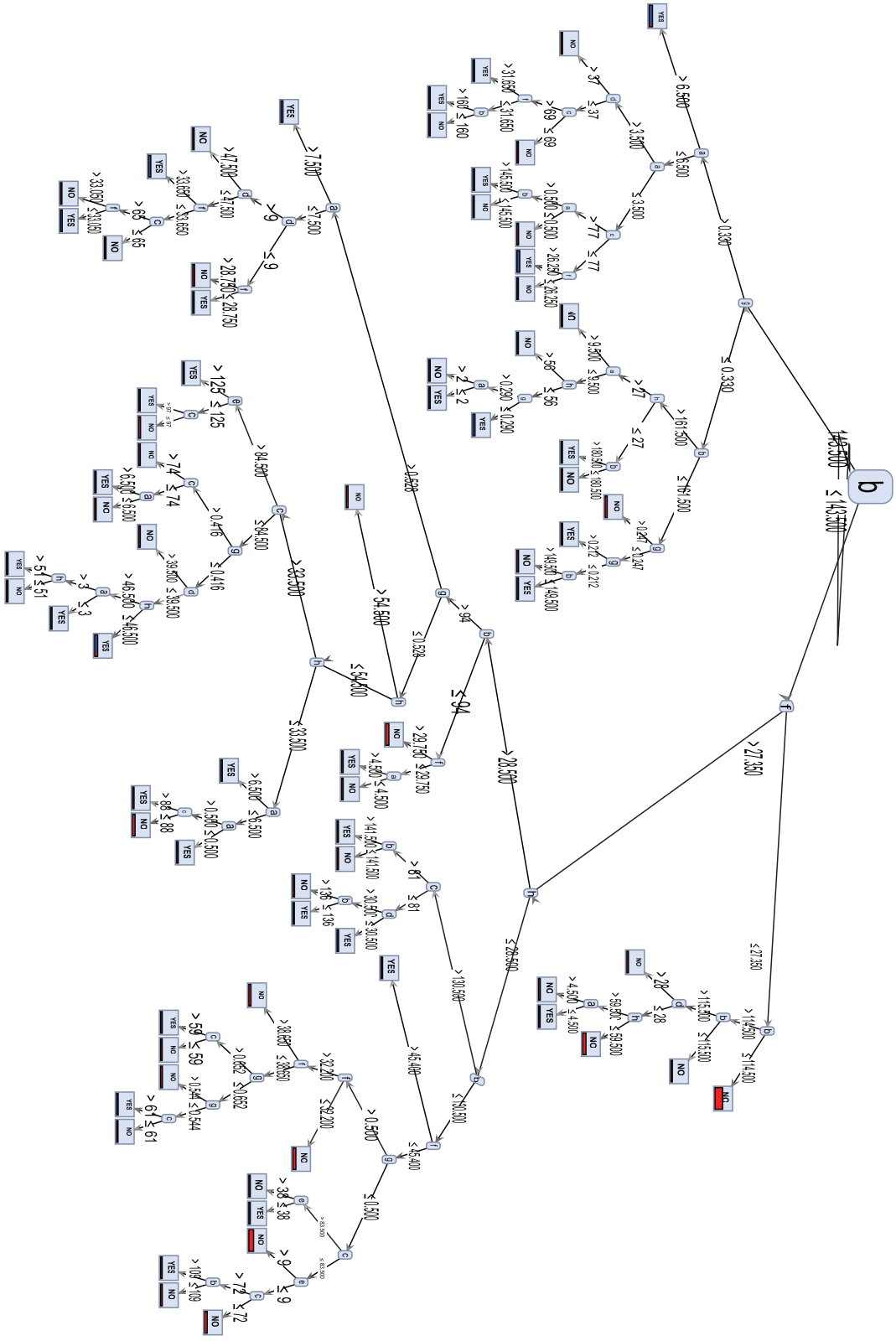

Figure 2. Decision tree for diabetes diagnosis.

In our study, all of the ANN-based classifications were done using the MATLAB Neural Network Toolbox.

PNNs are feedforward networks built with 3 layers. When an input is presented, the first layer calculates distances from the input vector to the training input vectors and produces a vector whose elements specify how close the input is to a training input. The second layer accumulates these contributions for each class of inputs to produce net output vector probabilities. They train quickly since the training is done in one pass of each training vector, rather than several passes [18,19].

LVQ neural networks consist of competitive and linear layers. The first layer maps input vectors into clusters. These are found by the network in the process of training. After this step, the network maps the merged groups of the first-layer clusters into the classes defined by the target data [20,21].

In FFNs, inputs are directly associated with outputs. The information travels in only one direction, from the input nodes through the hidden nodes (if any) and to the output nodes [22,23].

CFNs resemble FFNs. The difference in CFNs is that each layer's neurons relate to all of the previous layer's neurons. CFNs contain a weight connection from the input to each layer and from each layer to the successive layers [24,25].

DTDNs also resemble FFNs. The difference is that throughout DTDNs, each of the input and layer weights has a tap delay line associated with it. As a result of this, the network has a finite dynamic response to the time-series input data [26,27].

TDNs resemble FFNs, as well. The difference is that TDNs have a tapped delay line at the input weight, in which the dynamics appear only at the input layer of a static multilayer feedforward network. This allows the network to have a finite dynamic response to time-series input data [28,29].

2.5. ROC values

The results of each classifier were compared using ROC values. The ROC can be expressed as the fraction of true positives out of false positives [30,31]. True positive (TP), true negative (TN), false positive (FP), and false negative (FN) are the expressions used for the sensitivity, specificity, and accuracy parameters [31,32]. The definitions of these abbreviations in this study are given in Table 4.

Table 4. ROC parameters.

Abbreviation	Definition
TP	The number of people that the program found to have diabetes among the people diagnosed with diabetes by a specialist physician
TN	The number of people that the program found to be healthy among the people diagnosed as healthy by a specialist physician
FP	The number of people that the program found to have diabetes among the people diagnosed as healthy by a specialist physician
FN	The number of people that the program found to be healthy among the people diagnosed with diabetes by a specialist physician

Necessary parameter calculations for the ROC are given in Eqs. (5), (6), and (7) [31,32]:

$$\text{Sensitivity} = \frac{\text{TP}}{(\text{TP} + \text{FN})}, \tag{5}$$

$$\text{Specificity} = \frac{\text{TN}}{(\text{TN} + \text{FP})}, \tag{6}$$

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})}. \tag{7}$$

3. Results

In this section, the results for 8 different classifiers are given one by one in Tables 5–12. After each of these tables, the accuracy, sensitivity, and specificity values are given in the following paragraphs. These values were obtained from the results that are shown in Tables 5–12. Finally, all of the results are shown together in Table 13 and its graphical display is given in Figure 3.

Table 5. Confusion matrix for the PNN.=

TP = 57	FP = 37	94
FN = 33	TN = 123	156
90	160	250

The classification results that were made with the PNN are given in Table 5. Accordingly, out of 90 data points that the physician considered as positive, the PNN found that 57 were positive and 33 were negative. Moreover, out of 160 data that the physician considered as negative, the PNN found that 123 were negative and 37 were positive. Therefore, the PNN gave values of 72% accuracy, 63.33% sensitivity, and 76.88% specificity.

Table 6. Confusion matrix for the LVQ.

TP = 49	FP = 25	74
FN = 41	TN = 135	176
90	160	250

The classification results that were made with the LVQ are given in Table 6. Accordingly, out of 90 data that the physician considered as positive, the LVQ found that 49 were positive and 41 were negative. Moreover, out of 160 data that the physician considered as negative, the LVQ found that 135 were negative and 25 were positive. Therefore, the LVQ gave values of 73.6% accuracy, 54.44% sensitivity, and 84.38% specificity.

Table 7. Confusion matrix for the FFN.

TP = 49	FP = 37	86
FN = 41	TN = 123	164
90	160	250

The classification results that were made with the FFN are given in Table 7. Accordingly, out of 90 data that the physician considered as positive, the FFN found that 49 were positive and 41 were negative. Moreover, out of 160 data that the physician considered as negative, the FFN found that 123 were negative and 37 were positive. Therefore, the FFN gave values of 68.8% accuracy, 54.44% sensitivity, and 76.88% specificity.

Table 8. Confusion matrix for the CFN.

TP = 56	FP = 46	102
FN = 34	TN = 114	148
90	160	250

The classification results that were made with the CFN are given in Table 8. Accordingly, out of 90 data that the physician considered as positive, the CFN found that 56 were positive and 34 were negative. Moreover, out of 160 data that the physician considered as negative, the CFN found that 114 were negative and 46 were positive. Therefore, the CFN gave values of 68% accuracy, 62.22% sensitivity, and 71.25% specificity.

Table 9. Confusion matrix for the DTDN.

TP = 48	FP = 18	66
FN = 42	TN = 142	184
90	160	250

The classification results that were made with the DTDN are given in Table 9. Accordingly, out of 90 data that the physician considered as positive, the DTDN found that 48 were positive and 42 were negative. Moreover, out of 160 data that the physician considered as negative, the DTDN found that 142 were negative and 18 were positive. Therefore, the DTDN gave values of 76% accuracy, 53.33% sensitivity, and 88.75% specificity.

Table 10. Confusion matrix for the TDN.

TP = 37	FP = 30	67
FN = 53	TN = 130	183
90	160	250

The classification results that were made with the TDN are given in Table 10. Accordingly, out of 90 data that the physician considered as positive, the TDN found that 37 were positive and 53 were negative. Moreover, out of 160 data that the physician considered as negative, the TDN found that 130 were negative and 30 were positive. Therefore, the TDN gave values of 66.8% accuracy, 41.11% sensitivity, and 81.25% specificity.

Table 11. Confusion matrix for the Gini algorithm.

TP = 38	FP = 34	72
FN = 47	TN = 119	166
85	153	238

The classification results that were made with a decision tree using the Gini algorithm are given in Table 11. Accordingly, out of 90 data that the physician considered as positive, the decision tree found that 38 were positive and 47 were negative. Moreover, out of 160 data that the physician considered as negative, the decision tree found that 11 were negative and 34 were positive. The decision tree was not able to categorize 12 of the data, including 5 of the data that the physician had considered as positive and 7 of the data that the physician had considered as negative. Therefore, the decision tree gave values of 65.97% accuracy, 44.71% sensitivity, and 77.78% specificity.

Table 12. Confusion matrix for the AIS.

TP = 47	FP = 35	82
FN = 43	TN = 125	168
90	160	250

The classification results that were made with the AIS are given in Table 12. Accordingly, out of 90 data that the physician considered as positive, the AIS found that 47 were positive and 43 were negative. Moreover, out of 160 data that the physician considered as negative, the AIS found that 125 were negative and 35 were positive. Therefore, the AIS gave values of 68.8% accuracy, 52.22% sensitivity, and 78.13% specificity.

The achievement values for all 8 classifiers are given in Table 13 and the same results are shown in the graph in Figure 3.

Table 13. Comparison of results for the methods used. Best results are bolded.

	Accuracy (%)	Sensitivity (%)	Specificity (%)
PNN	72.00	**63.33**	76.88
LVQ	73.60	54.44	84.38
FFN	68.80	54.44	76.88
CFN	68.00	62.22	71.25
DTDN	**76.00**	53.33	**88.75**
TDN	66.80	41.11	81.25
GINI	65.97	44.71	77.78
AIS	68.80	52.22	78.13

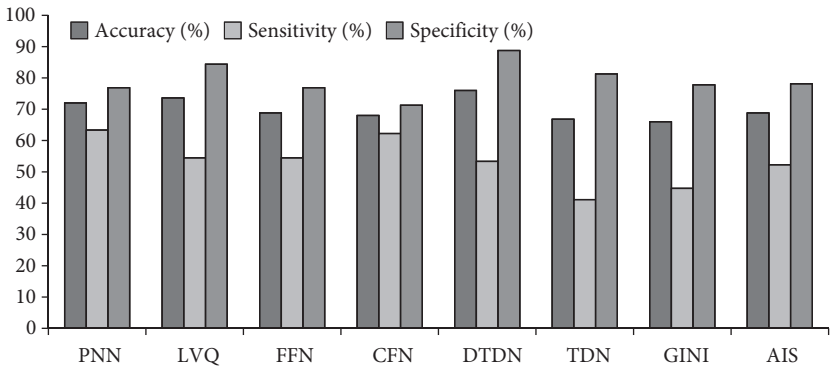

Figure 3. Graphical results of the classification.

This study and several other studies conducted using the same database are compared in Table 14. This table is examined in Section 4.

4. Conclusions

When this study was compared to other studies using the same database, it was seen that, in general, the accuracy values were close to each other. In this study, the accuracy values lagged a little bit behind compared to those in other studies. However, in the other studies shown in Table 14, sensitivity and specificity values were not given. Examining the results obtained in this study, it can be seen that the sensitivity and specificity values are also very important in order to evaluate the success of a classifier. In addition, different studies using the same classifier were able to achieve different performance rates. For example, Temurtas et al. [6] used a MLNN structure, which was trained by the LM algorithm, as Kayaer and Yildirim [7] used, but they achieved different performance rates. Again, Temurtas et al. used a PNN, as in this study, but they achieved different performance rates. These differences may result from the selected simulation sets and the network parameters that were used in the study. In order to better evaluate the performance differences and to better understand

the performances of the classifier, despite the use of the same algorithm, it would be beneficiary to give the specificity and sensitivity values in addition to the accuracy values.

Table 14. Comparison results of the other studies.

Author (year)	Method	Accuracy	Sensitivity	Specificity
This study (2014)	PNN	72.00%	63.33%	76.88%
	LVQ	73.60%	54.44%	84.38%
	FFN	68.80%	54.44%	76.88%
	CFN	68.00%	62.22%	71.25%
	DTDN	76.00%	53.33%	88.75%
	TDN	66.80%	41.11%	81.25%
	Gini	65.97%	44.71%	77.78%
	AIS	68.80%	52.22%	78.13%
Temurtas et al. (2009) [6]	MLNN with Levenberg-Marquardt (LM) (10xFC)	79.62%	Unspecified	Unspecified
	PNN (10xFC)	78.05%		
	MLNN with LM	82.37%		
	PNN	78.13%		
Kayaer and Yıldırım (2003) [7]	GRNN	80.21%	Unspecified	Unspecified
	MLNN with LM	77.08%		
Meng et al. (2005) [8]	AIRS	67.40%	Unspecified	Unspecified

When all of the classifiers in this study were compared, the best accuracy value was achieved with the DTDN, with a value of 76.00%; the best sensitivity value was achieved with the PNN, with a value of 63.33%; and the best specificity value was achieved with the DTDN, with a value of 88.75%. The second-best accuracy and specificity values after the DTDN were achieved with the LVQ network. The second-best performance for the sensitivity value was provided by the CFN. Since the correct identification of the patients was associated with sensitivity, practically, it was more convenient to use the PNN network, which indicated the best sensitivity performance.

References

[1] U. Masharani, J.H. Karam, Current Medical Diagnosis & Treatment 2003: A Lange Medical Book, San Francisco, CA, USA, McGraw-Hill, 2003.

[2] WHO, World Diabetes Statistics, Geneva, Switzerland, WHO, available at http://www.who.int/diabetes/en/index.html, ©2014.

[3] Y. Huang, P. McCullagh, N. Black, R. Harper, "Feature selection and classification model construction on type 2 diabetic patients data", Artificial Intelligence in Medicine, Vol. 41, pp. 251–262, 2007.

[4] S. Lekkas, L. Mikhailov, "Evolving fuzzy medical diagnosis of Pima Indians diabetes and of dermatological diseases", Artificial Intelligence in Medicine, Vol. 50, pp. 117–126, 2010.

[5] K. Polat, S. Güneş, "An expert system approach based on principal component analysis and adaptive neuro-fuzzy inference system to diagnosis of diabetes disease", Digital Signal Processing, Vol. 17, pp. 702–710, 2007.

[6] H. Temurtas, N. Yumusak, F. Temurtas, "A comparative study on diabetes disease diagnosis using neural networks", Expert Systems with Applications, Vol. 36, pp. 8610–8615, 2009.

[7] K. Kayaer, T. Yildirim, "Medical diagnosis on Pima Indian diabetes using general regression neural networks", Proceedings of the International Conference on Artificial Neural Networks and Neural Information Processing, pp. 181–184, 2003.

[8] L. Meng, P. Putten, H. Wang, "A comprehensive benchmark of the artificial immune recognition system (AIRS)", Proceedings of the First International Conference on Advanced Data Mining and Applications, pp. 575–582, 2005.

[9] New Mexico State University, The Human Genome Project and Diabetes: Genetics of Type II Diabetes, Las Cruces, NM, USA, 1997.

[10] UCI Machine Learning, Pima Indians Diabetes Data Set, Irvine, CA, USA, University of California Irvine, available at http://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes.

[11] O. Engin, A. Döyen, "Artificial immune systems and applications in industrial problems", Gazi University Journal of Science, Vol. 17, pp. 71–84, 2004.

[12] L.N. Castro, J. Timmis, Artificial Immune Systems: A New Computational Intelligence Approach, 1st ed., London, UK, Springer, 2002.

[13] S.M. Garrett, "How do we evaluate artificial immune systems", Evolutionary Computation, Vol. 13, pp. 145–178, 2005.

[14] K. Polat, S. Güneş, "Computer aided medical diagnosis system based on principal component analysis and artificial immune recognition system classifier algorithm", Expert System with Applications, Vol. 34, pp. 773–779, 2008.

[15] Y. Özkan, Veri Madenciliği Yöntemleri, İstanbul, Turkey, Papatya Yayıncılık, 2008 (in Turkish).

[16] Rapidminer, Rapid - I - Rapidminer, available at http://rapid-i.com/content/view/181/190.

[17] E. Öztemel, Artificial Neural Networks, İstanbul, Turkey, Papatya Yayıncılık, 2003.

[18] Probabilistic Neural Networks, available at http://herselfsai.com/2007/03/probabilistic-neural-networks.html.

[19] MathWorks, Probabilistic Neural Networks Radial Basis Networks, available at http://www.mathworks.com/help/toolbox/nnet/ug/bss38ji-1.html.

[20] Learning Vector Quantization Networks – Self-Organizing and Learning Vector Quantization Nets, available at http://www.kxcad.net/cae_MATLAB/toolbox/nnet/self_o17.html.

[21] MathWorks, Learning Vector Quantization Neural Network - MATLAB, available at http://www.mathworks.com/help/toolbox/nnet/ref/lvqnet.html.

[22] Wikipedia, Feedforward Neural Network, available at http://en.wikipedia.org/wiki/Feedforward_neural_network.

[23] MathWorks, Feedforward Neural Network – MATLAB, available at http://www.mathworks.com/ help/toolbox/nnet/ref/feedforwardnet.html.

[24] Feedforward Network Backpropagation (Neural Network Toolbox), available at http://www.kxcad.net/cae_MATLAB/toolbox/nnet/backpro7.html.

[25] MathWorks, Cascade Forward Neural Network – MATLAB, available at http://www.mathworks.com/ help/toolbox/nnet/ref/cascadeforwardnet.html.

[26] MathWorks, Distributed Time Delay Neural Network – Dynamic Networks (Neural Network Toolbox), available at http://www.mathworks.com/help/toolbox/nnet/ug/bss36c8-1.html.

[27] MathWorks, Distributed Delay Network – MATLAB, available at http://www.mathworks.com/help/toolbox/ nnet/ref/distdelaynet.html.

[28] Focused Time Delay Neural Network – Dynamic Networks (Neural Network Toolbox), available at http://www.kxcad.net/cae_MATLAB/toolbox/nnet/dynamic7.html.

[29] MathWorks, Time Delay Neural Network – MATLAB, available at http://www.mathworks.com/help/toolbox/nnet/ref/timedelaynet.html.

[30] Wikipedia, ROC, available at http://tr.wikipedia.org/wiki/ROC (in Turkish).

[31] Wikipedia, Receiver operating characteristic, available at http://en.wikipedia.org/wiki/Receiver_operating_characteristic.

[32] K.K. Gündoğan, B. Alataş, A. Karcı, "Mining classification rules by using genetic algorithms with non-random initial population and uniform operator", Turkish Journal of Electrical Engineering & Computer Science, Vol. 12, pp. 43–52, 2004.