International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 4 | ISSN : 2456-3307

Three-Fold Integrated Clutsering-Classification (TICC) Strategy for Diabetes Mellitus Prediction

Dr. V. Saravanan[1], Monika Seles[2]

[1]Associate Professor & Head, PG and Research Department of Information Technology, Hindusthan College of Arts and Science, Coimbatore, Tamil Nadu, India
[2]M. Phil Research Scholar, Hindusthan College of Arts and Science, Coimbatore, Tamil Nadu, India

ABSTRACT

MLT finds potentially useful patterns in the data. Three MLT deployed for the diabetes mellitus prediction is presented subsequently with a brief on proposed method, experimental set up, test results and performance comparison. The proposed classifiers are tested with the original dataset. The results are recorded first. Subsequently the dataset will be subject to cluster and the this will be the first fold of the proposed technique. In the expansion step the assigned cluster will be a separate instance in the dataset. This will be the second fold of the proposed technique. Classification will be deployed as the third fold of the proposed technique. This proposed three fold integrated clustering-classification technique for diabetes mellitus prediction significantly improves the performance of the diabetes mellitus prediction. After the proposed strategy is carried out, results are recorded and compared.

Keywords : MLT, Diabetes Mellitus, Classification, Clustering

I. INTRODUCTION

Researchers proposed many methods for diabetes mellitus prediction using MLT. Diabetes-Mellitus refers to the metabolic disorder that happens from misfunction in insulin secretion and action. It is characterized by hyperglycemia. The persistent hyperglycemia of diabetes leads to damage, malfunction and failure of different organs such as kidneys, eyes, nerves, blood vessels and heart. In the past decades several techniques have been implemented for the detection of diabetes.

The diagnosis of diabetes mellitus is very important now a days using various types of techniques. Here, there are various techniques, their classification and implementation using various types of software tools and techniques. The diagnosis of diabetes can be done using Artificial Neural Network, K-fold cross validation and classification, Vector support machine, K-nearest neighbor method, Data Mining Algorithm, etc. This work utilizes three methods for the problem:

- Voted Perceptron
- Multilayer Perceptron
- Bayesnet classification

II. Technical Perspetives Of Working Methodology

Machine learning deals with the erection and study of systems that learns from data. A training dataset with its corresponding feature vectors and labels are fed into the machine learners. Prediction is made for the test dataset and expected class is determined (Figure 4.1). In diabetes mellitus prediction, samples are mapped either TESTED-POSITIVE OR TESTED-NEGATIVE.

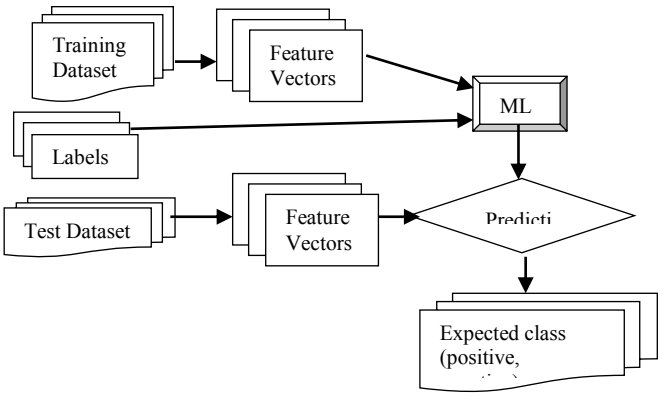


Figure 4.1: Machine Learning Techniques Working Scenario

This experiment have done with the help of open source data mining tools in window environment using net beans software. In this experiment, we have used java code and libraries which are available in WEKA. Machine learning methods such as neural networks and decision tree methods have been deployed for the problem as witnessed from the literature. Three methods which are not commonly deployed yet for the problem is focused in this research:

- Voted Perceptron
- Multilayer Perceptron
- Bayesnet classification

Experimental setup, test results, performance comparison are explained in the subsequent Sections.

III. Evaluation Metrics

Machine Learning performance efficiency is evaluated with the metrics such as Precision, Recall and F-Measure. The total samples are divided into True Positives (TP), False Positives (FP), False Negatives (FN) and True Negatives (TN). Consider Positive (identified) and Negative (rejected), then

- True Positive: Number of correctly identified samples
- False Positive: Number of incorrectly identified samples
- True Negative: Number of correctly rejected samples
- False Negative: Number of incorrectly rejected
- samples

The evaluation metrics with their appropriate formulas are enlisted in Table 4.2. Experiments conducted in the work deploy these metrics.

Table 4.1: Evaluation Metrics

Confusion Matrix		Actual outcome (Observation)		
		Positive	Negative	
Test outcome (Expectation)	Positive	*True Positive* (TP)	*False Positive* (FP)	Positive Predictive Value (PPV) (or) Precision (α) = TP/(TP+FP)
	Negative	*False Negative* (FN)	*True Negative* (TN)	Negative Predictive Value (NPV) = TN/(TN+FN)
		Sensitivity or Recall (β) = TP/(TP+FN)	Specificity (or) True Negative Value (TNV) = TN/(TN+FP)	Accuracy (ACC) = (TP+TN)/(TP + FP+TN+FN)
				F-Score = 2.(α. β)/(α+ β)

IV. Multilayer perceptron for diabetes mellitus prediction

Multilayer Perceptron Classifier uses backpropagation to classify instances. This network can be built manually, created by an algorithm or both. The network can also be monitored and modified during training time. The nodes in this network are all sigmoid (except for when the class is numeric in which case the output nodes become unthresholded linear units).

4.4.1 Experimental Setup

The settings deployed in MultilayerPerceptron for the Diabetes Mellitus is as follows:

- ✓ autoBuild -- Adds and connects up hidden layers in the network.
- ✓ debug -- If set to true, classifier may output additional info to the console.
- ✓ decay -- This will cause the learning rate to decrease. This will divide the starting learning rate by the epoch number, to determine what the current learning rate should be. This may help to stop the network from diverging from the target output, as well as improve general performance. Note that the decaying learning rate will not be shown in the gui, only the original learning rate. If the learning rate is changed in the gui, this is treated as the starting learning rate.
- ✓ hiddenLayers -- This defines the hidden layers of the neural network. This is a list of positive whole numbers. 1 for each hidden layer. Comma seperated. To have no hidden layers put a single 0 here. This will only be used if autobuild is set. There are also wildcard values 'a' = (attribs + classes) / 2, 'i' = attribs, 'o' = classes , 't' = attribs + classes.
- ✓ learningRate -- The amount the weights are updated.
- ✓ momentum -- Momentum applied to the weights during updating.
- ✓ nominalToBinaryFilter -- This will preprocess the instances with the filter. This could help improve performance if there are nominal attributes in the data.
- ✓ normalizeAttributes -- This will normalize the attributes. This could help improve performance of the network. This is not reliant on the class being numeric. This will also normalize nominal attributes as well (after they have been run through the nominal to binary filter if that is in use) so that the nominal values are between -1 and 1
- ✓ normalizeNumericClass -- This will normalize the class if it's numeric. This could help improve performance of the network, It normalizes the class to be between -1 and 1. Note that this is only internally, the output will be scaled back to the original range.
- ✓ reset -- This will allow the network to reset with a lower learning rate. If the network diverges from the answer this will automatically reset the network with a lower learning rate and begin training again. This option is only available if the gui is not set. Note that if the network diverges but isn't allowed to reset it will fail the training process and return an error message.
- ✓ seed -- Seed used to initialise the random number generator.Random numbers are used for setting the initial weights of the connections betweem nodes, and also for shuffling the training data.
- ✓ trainingTime -- The number of epochs to train through. If the validation set is non-zero then it can terminate the network early
- ✓ validationSetSize -- The percentage size of the validation set.(The training will continue until it is observed that the error on the validation set has been consistently getting worse, or if the training time is reached).
- ✓ If This is set to zero no validation set will be used and instead the network will train for the specified number of epochs.
- ✓ validationThreshold -- Used to terminate validation testing.The value here dictates how many times in a row the validation set error can get worse before training is terminated.

4.4.2 Results

The results achieved for the MultilayerPerceptron for the problem of diabetes mellitus prediction is as follows:

Classification alone method results for MultiLayerPerceptron

Time taken to build model: 1.08 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	579	75.3906 %
Incorrectly Classified Instances	189	24.6094 %
Kappa statistic	0.4484	
Mean absolute error	0.2955	
Root mean squared error	0.4215	
Relative absolute error	65.0135 %	
Root relative squared error	88.4274 %	
Coverage of cases (0.95 level)	96.224 %	
Mean rel. region size (0.95 level)	84.375 %	
Total Number of Instances	768	

Table 4.3: MultiLayer Perceptron Confusion Matrix

=== Confusion Matrix ===	
a b	<-- classified as
416 84 |	a = tested_negative
105 163 |	b = tested_positive

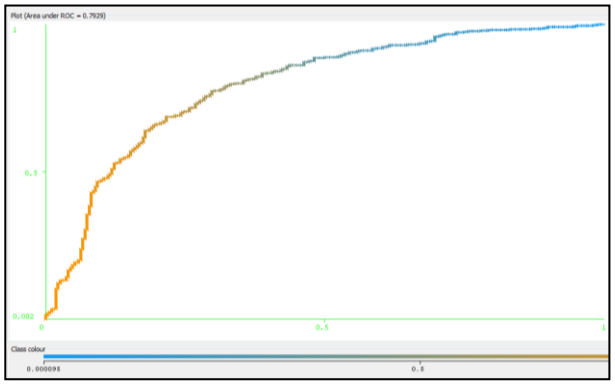


Figure 4.2: MultiLayerPerceptron ROC Curve (ROC Value = 0.7929)

Proposed Three Fold Integrated Clustering-Classification Results Of The Multilayer Perceptron

Time taken to build model: 1.2 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	765	99.6094 %
Incorrectly Classified Instances	3	0.3906 %
Kappa statistic	0.9911	
Mean absolute error	0.0084	
Root mean squared error	0.0658	
Relative absolute error	1.9059 %	
Root relative squared error	13.994 %	
Coverage of cases (0.95 level)	100 %	
Mean rel. region size (0.95 level)	51.3021 %	
Total Number of Instances	768	

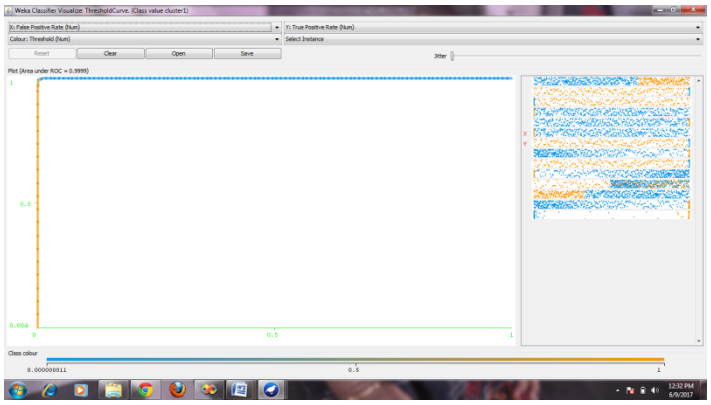


Figure 4.3: Proposed TICC MultiLayerPerceptron ROC Curve (ROC Value = 0.99)

The proposed method shows the drastic improvement for utmost 20% in ROC value. The strategy may help in improving the accuracy of the prediction of the diabetes mellitus and which can further works well in diagnosis.

Votedperceptron for Diabetes Mellitus

The voted perceptron method is based on the perceptron algorithm of Rosenblatt and Frank. The algorithm takes advantage of data that are linearly separable with large margins. This method is simpler to implement, and much more efficient in terms of computation time as compared to Vapnik's SVM. The algorithm can also be used in very high dimensional spaces using kernel functions.

4.5.1 Experimental Setup

The parameters used in the VotedPerceptron is as follows:

- debug -- If set to true, classifier may output additional info to the console.
- exponent -- Exponent for the polynomial kernel.
- maxK -- The maximum number of alterations to the perceptron.
- numIterations -- Number of iterations to be performed.
- seed -- Seed for the random number generator.

4.5.2 Results

Classification alone method results for VotedPerceptron

Time taken to build model: 0.13 seconds
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	513 66.7969 %
Incorrectly Classified Instances	255 33.2031 %
Kappa statistic	0.1353
Mean absolute error	0.3319
Root mean squared error	0.5752
Relative absolute error	73.0209 %
Root relative squared error	120.6751 %
Coverage of cases (0.95 level)	67.0573 %
Mean rel. region size (0.95 level)	50.1953 %
Total Number of Instances	768

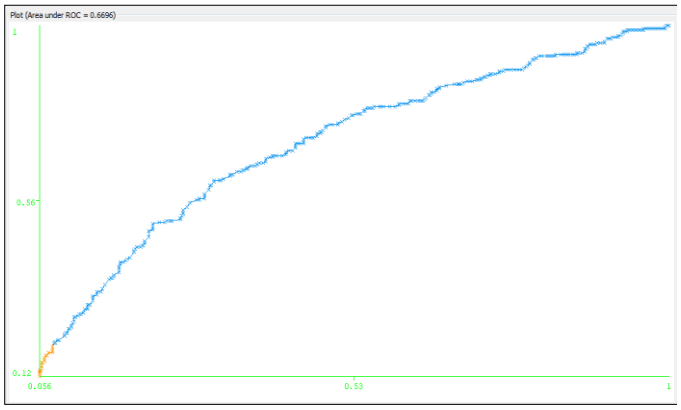


Figure 4.4: VotedPerceptron ROC Curve (ROC Value = 0.6696)

Proposed Three Fold Integrated Clustering-Classification Results Of The VotedPerceptron

Time taken to build model: 0.08 seconds
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	*546 71.0938 %*
Incorrectly Classified Instances	*222 28.9063 %*
Kappa statistic	*0.2031*
Mean absolute error	*0.2891*
Root mean squared error	*0.5372*
Relative absolute error	*65.4011 %*
Root relative squared error	*114.2879 %*
Coverage of cases (0.95 level)	*71.224 %*
Mean rel. region size (0.95 level)	*50.1302 %*
Total Number of Instances	*768*

Table 4.8: Proposed TICC VotedPerceptron Confusion Matrix

=== Confusion Matrix ===
a b <-- classified as
493 22 | a = cluster0
200 53 | b = cluster1

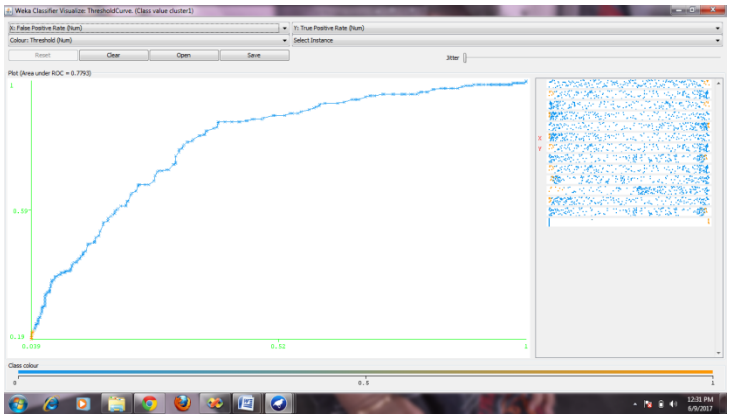


Figure 4.5: Proposed TICC VotedPerceptron ROC Curve (ROC Value = 0.6696)

By deploying the proposed TICC strategy in VotedPerceptron, the ROC value improves by 4%, this in turn helps to increase the diagnosis of the diabetes mellitus prediction problem.

Bayesnet classifier for diabetes mellitus prediction

A Bayesian network, Bayes network, belief network, Bayes(ian) model or probabilistic directed acyclic graphical model is a probabilistic graphical model (a type of statistical model) that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases.

Suppose that there are two events which could cause grass to be wet: either the sprinkler is on or it's raining. Also, suppose that the rain has a direct effect on the use of the sprinkler (namely that when it rains, the sprinkler is usually not turned on). Then the situation can be modeled with a Bayesian network (shown to the right). All three variables have two possible values, T (for true) and F (for false).

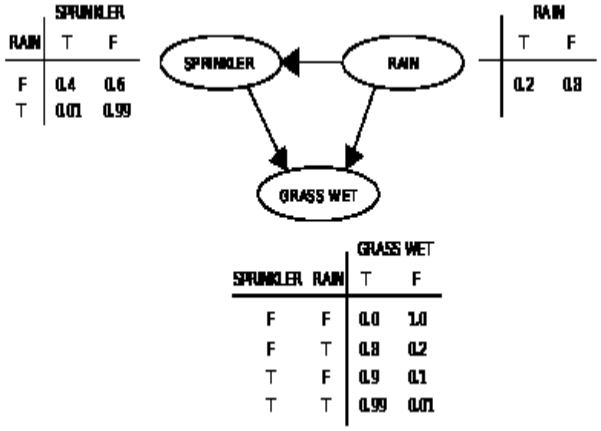


Figure 4.6: A simple Bayesian network with conditional probability tables

Bayes Network learning using various search algorithms and quality measures. Base class for a Bayes Network classifier. Provides datastructures (network structure, conditional probability distributions, etc.) and facilities common to Bayes Network learning algorithms like K2 and B.

4.6.1 Experimental Setup

The parameters used in Bayesnet classifier is as follows:

- BIFFile -- Set the name of a file in BIF XML format. A Bayes network learned from data can be compared with the Bayes network represented by the BIF file. Statistics calculated are o.a. the number of missing and extra arcs.
- debug -- If set to true, classifier may output additional info to the console.
- estimator -- Select Estimator algorithm for finding the conditional probability tables of the Bayes Network.
- searchAlgorithm -- Select method used for searching network structures.
- useADTree -- When ADTree (the data structure for increasing speed on counts, not to be confused with the classifier under the same name) is used learning time goes down typically. However, because ADTrees are memory intensive, memory problems may occur. Switching this option off makes the structure learning algorithms slower, and run with less memory. By default, ADTrees are used.

4.6.2 Results

Classification alone method results for BayesNet

The results achieved in the Bayesnet Classifier is as follows:

Time taken to build model: 0.05 seconds
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 571 74.349 %
Incorrectly Classified Instances 197 25.651 %
Kappa statistic 0.429
Mean absolute error 0.2987
Root mean squared error 0.4208
Relative absolute error 65.7116 %
Root relative squared error 88.28 %
Coverage of cases (0.95 level) 97.1354 %
Mean rel. region size (0.95 level) 84.2448 %
Total Number of Instances 768

4.6.3 Proposed Three Fold Integrated Clustering-Classification Results Of The BayesNet

Time taken to build model: 0.03 seconds
=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	*732*
95.3125 %	
Incorrectly Classified Instances	*36*
4.6875 %	
Kappa statistic	*0.8966*
Mean absolute error	*0.0658*
Root mean squared error	*0.1821*
Relative absolute error	*14.8949 %*
Root relative squared error	*38.7495 %*
Coverage of cases (0.95 level)	*99.8698 %*
Mean rel. region size (0.95 level)	*59.1146 %*
Total Number of Instances	*768*

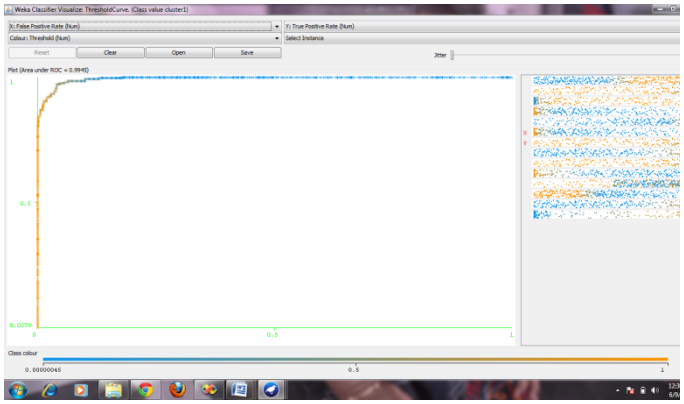


Figure 4.8: Proposed TICC BayesNet ROC Curve (ROC Value = 0.994)

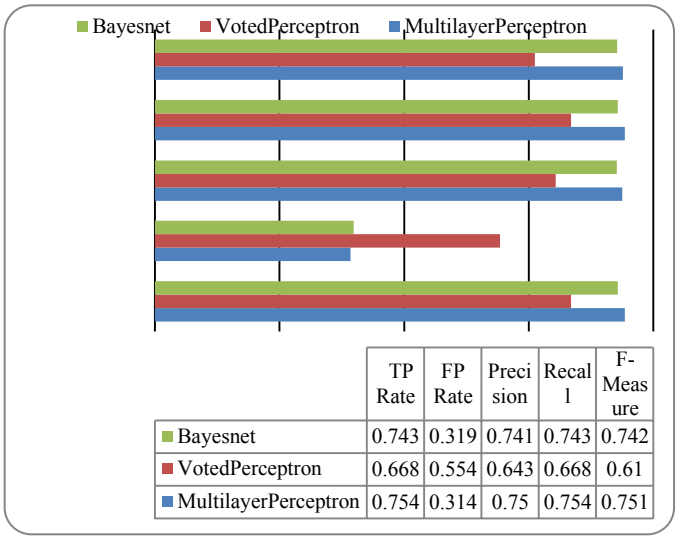
The proposed TICC strategy when adopted the performance is improved by 20% in BayesNet Classifier. Improving accuracy by adopting certain techniques would help in diagnosing the diabetes mellitus in early stages rather than finding it in advanced stages.

V. Findings And Discussion

Out The proposed classifiers are tested with the original dataset and recorded results seems to be moderate in performance. Subsequently when the dataset is subject to cluster as first fold of the proposed TICC strategy, results seems to be well tuned with hiked performance. In the expansion step the assigned cluster acts as a separate instance in the dataset. This helps for further improve the performance of the proposed technique.

Classification deployed as the third fold of the proposed technique seems to provide drastic improvement in overall performance optimization. This proposed three fold integrated clustering-classification technique for diabetes mellitus prediction significantly improves the performance of the diabetes mellitus prediction. After the proposed strategy is carried out, results are recorded and compared as shown in Figure 4.9 and Figure 4.10. Among the three methods for the problem deployed in the experiments the performance hike is as follows:

- Voted Perceptron – 4%
- Multilayer Perceptron – 19%
- Bayesnet classification – 20%



	TP Rate	FP Rate	Precision	Recall	F-Measure
Bayesnet	0.743	0.319	0.741	0.743	0.742
VotedPerceptron	0.668	0.554	0.643	0.668	0.61
MultilayerPerceptron	0.754	0.314	0.75	0.754	0.751

Figure 4.9 Performance Comparison

Hence, the BayesNet algorithm yields good performance for the diabetes mellitus problem, it is implemented in MATLAB. The BayesNet network created for the diabetes mellitus prediction will be explored in subsequent paper.

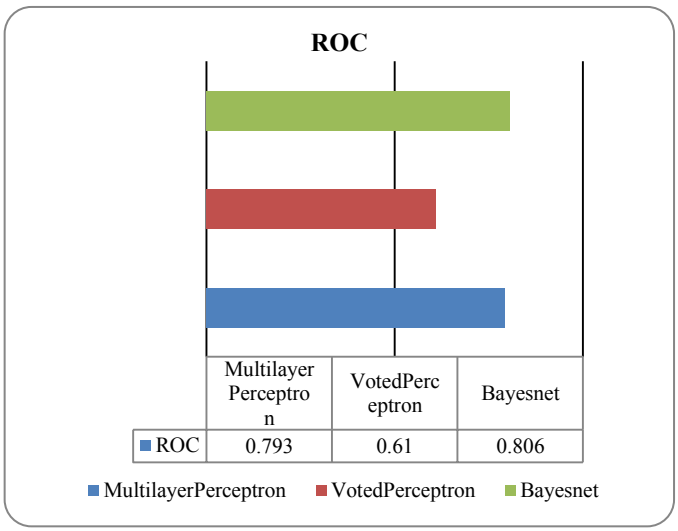


Figure 4.10 ROC Comparison without TICC strategy

The etiology of diabetes in India is multifactorial and includes genetic factors coupled with environmental influences such as obesity associated with rising living standards, steady urban migration, and lifestyle changes. Yet despite the incidence of diabetes within India, there are no nationwide and few multi-centric studies conducted on the prevalence of diabetes and its complications. The studies that have been undertaken are also prone to potential error as the heterogeneity of the Indian population with respect to culture, ethnicity, socio- economic conditions, mean that the extrapolation of regional results may give inaccurate estimates for the whole country.

An upsurge in number of early-onset diabetes cases is also responsible for the development of various diabetic complications due to longer disease duration, however data on the prevalence on diabetic complications across the whole of India is scarce. A recent international study reported that diabetes control in individuals worsened with longer duration of the disease (9.9±5.5 years), with neuropathy the most common complication (24.6 per cent) followed by cardiovascular complications (23.6 per cent), renal issues (21.1 per cent), retinopathy (16.6 per cent) and foot ulcers (5.5 per cent).

These results were closely in line with other results from the South Indian population, however further data from different sections of India is required to be able to assess whether patterns of complications rates vary across the country. Poor glycaemic control, a factor that has been observed in the Indian diabetic population, is responsible for micro- and macrovascular changes that present with diabetes, and can predispose diabetic patients to other complications such as diabetic myonecrosis and muscle infarction.

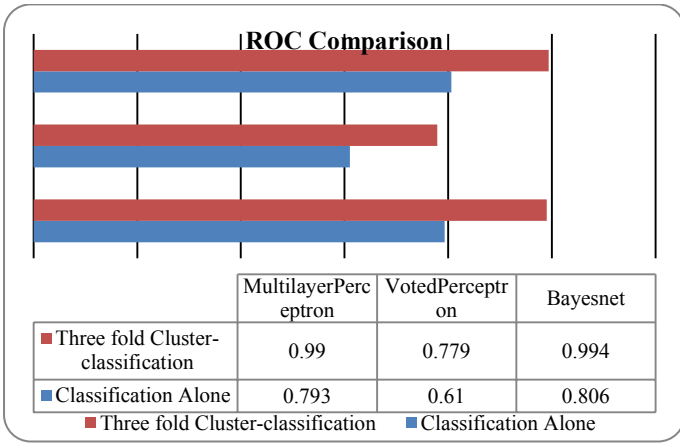


Figure 4.11 ROC Comparison without and with TICC strategy

VI. CONCLUSION

Addressing the diabetes mellitus seems to be most important aspect as of witnessed from these reports. The proposed TICC strategy adopted classifiers seems to perform well for the diabetes mellitus prediction. This strategy hikes and optimizes the overall performance. Among the experimented classifiers the BayesNet seems to vary apt for the problem. Hence, it is clearly evident that BayesNet classifier with the proposed TICC based strategy seems to be optimal for the diabetes mellitus prediction problem.

VII. REFERENCES

[1]. Raj Anand, Vishnu Pratap Singh Kirar, Kavita Burse, " K-Fold Cross Validation and Classification Accuracy of PIMA Indian Diabetes Data Set Using Higher Order Neural Network and PCA ", IJSCE, Volume-2, Issue-6, January 2013, pp. 436-438, ISSN: 2231-2307.

[2]. Y. Angeline Christobel, P.Sivaprakasam, "A New Classwise k Nearest Neighbor (CKNN) Method for the Classification of Diabetes Dataset", IJEAT, Volume-2, Issue-3, February 2013, pp. 396-400, ISSN: 2249 – 8958.

[3]. Pujari A. K. et al. (2012). Improving Classification Accuracy by Using Feature Selection and Ensemble Model. International Journal of Soft Computing and Engineering (IJSCE). International Journal of Soft Computing and Engineering (IJSCE)Vol. 2,pp. 380-386.

[4]. Han, J.,& Micheline, K. (2006). Data mining: Concepts and Techniques, Morgan Kaufmann .Publisher.

[5]. UCI Repository of Machine Learning Databases, University of California at Irvine, Department of Computer Science. Available: http://www.ics.uci.edu/~mlearn/databases/thyroid -disease/newthyroid.data

[6]. Bishop, C. M. (1995). Neural Networks for Pattern Recognition. Oxford University Press, New York.

[7]. Bolli, G. (2006). Glucose variability and complications. Diabetes Care, 29(7):1707–1709.Box, G., Jenkins, G., and Reinsel, G. (2008). Time Series Analysis: Forecasting and Control. John Wiley, Hoboken, New Jersey, fourth edition.

[8]. Burges, C. (1998). A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2(2):121–167.
[9]. Cao, L. and Tay, F. (2003). Support vector machine with adaptive parameters in financial time series forecasting. IEEE Transactions on Neural Networks, 14(6):1506–1518.
[10]. "Diagnosis & Classification of Diabetes Mellitus", Diabetes Care, Volume 37, Supplement 1, 2014, pp. S81-S90.

VIII. Appendix – A

Sample Dataset

Database description

% 1. Title: Pima Indians Diabetes Database
% 5. Number of Instances: 768
% 6. Number of Attributes: 8 plus class
% 7. For Each Attribute: (all numeric-valued)
% 1. Number of times pregnant
% 2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test
% 3. Diastolic blood pressure (mm Hg)
% 4. Triceps skin fold thickness (mm)
% 5. 2-Hour serum insulin (mu U/ml)
% 6. Body mass index (weight in kg/(height in m)^2)
% 7. Diabetes pedigree function
% 8. Age (years)
% 9. Class variable (0 or 1)
% 8. Missing Attribute Values: None
% 9. Class Distribution: (class value 1 is interpreted as "tested positive for
% diabetes")
% Class Value Number of instances
% 0 500
% 1 268
@relation pima_diabetes
@attribute 'preg' real
@attribute 'plas' real
@attribute 'pres' real
@attribute 'skin' real
@attribute 'insu' real
@attribute 'mass' real
@attribute 'pedi' real
@attribute 'age' real
@attribute 'class' { tested_negative, tested_positive}
@data
6,148,72,35,0,33.6,0.627,50,tested_positive
1,85,66,29,0,26.6,0.351,31,tested_negative
8,183,64,0,0,23.3,0.672,32,tested_positive
1,89,66,23,94,28.1,0.167,21,tested_negative
0,137,40,35,168,43.1,2.288,33,tested_positive
5,116,74,0,0,25.6,0.201,30,tested_negative
3,78,50,32,88,31,0.248,26,tested_positive
10,115,0,0,0,35.3,0.134,29,tested_negative
2,197,70,45,543,30.5,0.158,53,tested_positive
8,125,96,0,0,0,0.232,54,tested_positive
4,110,92,0,0,37.6,0.191,30,tested_negative
10,168,74,0,0,38,0.537,34,tested_positive
10,139,80,0,0,27.1,1.441,57,tested_negative
1,189,60,23,846,30.1,0.398,59,tested_positive