EAI Endorsed Transactions
on Scalable Information Systems
Research Article EAI.EU

A Novel Approach for Prediction of Gestational Diabetes based on Clinical Signs and Risk Factors

Shiva Shankar Reddy[1,*], Mahesh Gadiraju[1], N. Meghana Preethi[1], V.V.R.Maheswara Rao[2]

[1]Department of CSE, Sagi Rama Krishnam Raju Engineering College (A), Bhimavaram, Andhrapradesh, INDIA
[2]Department. of CSE, Shri Vishnu Engineering College for Women (A), Bhimavaram, Andhrapradesh, INDIA

Abstract

Gestational diabetes mellitus occurs due to high glucose levels in the blood. Pregnant women are affected by this type of diabetes. A blood test is to be performed to identify diabetes. The Oral Glucose Tolerance Test (OGTT) is a blood test performed between the 24th and 28th week of pregnancy that is necessary to identify and overcome the side effects of GDM. The main objective of this work is to train a model by utilizing the training data, evaluate the trained model using the test data, and compare existing machine learning algorithms with a Gradient boosting machine (GBM) to achieve a better model for the effective prediction of gestational diabetes. In this work, the analysis was done with a few existing algorithms and the Extreme learning machine and Gradient boosting techniques. The k-fold cross-validation technique is applied with values of k as 3, 5, and 10 to obtain better performance. The existing algorithms implemented are the Naive Bayes classifier, Support Vector Machine, K-Nearest Neighbour, ID3, CART and J48. The proposed algorithms are Gradient boosting and ELM. These algorithms are implemented in R programming. The metrics like accuracy, kappa statistic, sensitivity/Recall, specificity, precision, f-measure and AUC are used to compare all the algorithms. GBM has obtained better performance than existing algorithms. Then finally, GBM is compared with the other proposed robust Machine Learning algorithm, namely the Extreme learning machine, and the GBM performed better. So, It is recommended to use a gradient-boosting algorithm to predict gestational diabetes effectively.

Keywords: Gestational diabetes mellitus (GDM), Naive Bayes (NB) classifier, Support Vector Machine, K-Nearest Neighbour (KNN), ID3, CART, J48,k-fold cross-validation, Extreme learning machine (ELM).

Received on 14 September 2022, accepted on 12 December 2022, published on 11 January 2023

doi: 10.4108/eetsis.v10i3.2697

*Corresponding author. Email: shiva.shankar591@gmail.com

1. Introduction

GDM occurs in pregnant women with high blood glucose levels due to insufficient insulin production by the pancreas. Every GDM patient must take proper treatment to avoid complications. Specific measures like physical exercises are to be taken to control gestational diabetes. The chance of affecting GDM is high if a pregnant woman has a previous history of GDM or obesity, or pre-diabetes [1]. A pregnant woman can identify the symptoms of gestational diabetes, like polyuria, polydipsia, polyphagia, and blurred vision. Every pregnant woman must take a proper test to detect diabetes in the early stage [2]. Long time suffering from GDM results in many complications like miscarriage, preterm delivery, excessive birth weight, nephropathy, high blood pressure, future type-2 diabetes and cardiovascular disease. GDM affects the growth of the foetus during pregnancy [3].

If GDM remains untreated for a long time, it may lead to cesarean delivery and affect the birth baby's organs. The baby may suffer from breathing disorders [4]. GDM can be prevented if the patient follows a simple routine like maintaining a healthy diet, ideal body weight, and regular physical exercise.

Early detection of GDM is necessary to avoid complications in future. GDM can be detected by performing blood tests regularly. The standard type of blood test performed is the OGTT. In OGTT, blood samples are routinely taken before and after breakfast. If the range of blood glucose levels is abnormal, the patient is said to have gestational diabetes [5].

Every doctor advises pregnant women to undergo an OGTT blood test between 24 and 28 weeks. GDM can be controlled by taking a proper diet initially. If the woman is highly affected by GDM, the doctor will give insulin to the patient. The insulin is induced through injection to maintain balanced blood sugar levels.

In machine learning, several classification algorithms were implemented on any specific dataset. The classification algorithms were used to classify data into different classes to train a model and to predict new instances. These predictions are evaluated with actual values to know how accurately the trained model predicts [6]. The Dataset used must be feasible for implementing algorithms. Data preprocessing is a technique applied to the Dataset to convert raw data into a possible and clean Dataset [7]. The re-sampling techniques, model training, testing, and evaluation are performed on the preprocessed data to obtain the best machine-learning model.

The re-sampling technique called cross-validation evaluates the trained model on a limited Dataset. There are several cross-validation techniques in machine learning. These techniques include Leave One Out Cross Validation (LOOCV), K-fold, Stratified cross-validation and Time-series cross-validation. In all these techniques, the Dataset is divided into training and test datasets [8].

In this work, ML classification algorithms predict gestational diabetes. Some prevalent and two considered techniques are executed in R. The existing algorithms are the NB classifier, SVM, K-NN, ID3, CART, ELM and J48. The proposed algorithm is an ensemble technique called Gradient boosting and an Extreme learning machine. These techniques are evaluated using k-fold cross-validation. In Section 3, the system architecture of the model is provided in detail.

2. Related work

Appaji Sangapu Venkata et al. [9] proposed their work on classifying cardiotocography class status. They used ML algorithms based on uterine contraction (UC) data and fetal heart rate (FHT) signals. They concluded that experimental results had shown good accuracy and a low error rate.

Chandra SekharVasamsetty et al. [10] analyzed gene expression for Type-2 Diabetes with parental history and healthy. They used some statistical methods like Mahalanobis distance, Minimum co-variance determinant, and less normalization to identify multivariate and univariate outliers. They applied these methods to microarray data. Their experimental results concluded that 1579 genes are differentially expressed out of 39400 genes. They performed biological process, molecular function and cellular component analysis using Gene Ontology and pathway analysis on 1579 differentially expressed genes.

Chandra SekharVasamsetty et al. [11] focused on identifying and classifying genes that cause Type-2 Diabetes with and without parental history. They used two statistical methods, Mahalanobis distance and Minimum co-variance determinant, to identify multivariate and univariate outliers. They performed functional classification using Gene Ontology and pathway analysis for identified inflammatory genes. Their experimental results concluded that 38 genes are differentially expressed out of 39400.

Geetha and Jayaveeran [12] highlighted their work on analyzing data mining techniques for predicting gestational diabetes. They used the K-means clustering algorithm and Decision table, Multilayer perceptron and NB classifier algorithms. After implementing the K-means clustering algorithm in the WEKA tool, they implemented classification algorithms. The Dataset consists of 5 attributes related to Gestational Diabetes. They compared classifiers based on performance measures like MSE, RMSE, RAE and RRSE. From the results obtained NB classifier performs better than other classifiers.

Jaya Mala [13] highlighted her work in predicting diabetes using classification techniques and feature selection. She considered the Pima dataset with 9 attributes and 768 instances. The proposed work implements SVM, J48 and NB classifiers using the WEKA tool. By comparing the performance measures of these three algorithms, she concluded that SVM with feature selection had obtained better accuracy of 78.30%.

Murat Koklu and YavuzUnal [14] mainly focused on three algorithms, NB, Multilayer perceptron and J48, to diagnose diabetes. They implemented these algorithms on PIMA Indian diabetes dataset. They compared the obtained results with results from previous studies. They concluded that the NB classifier achieved better accuracy of 76.302%.

Mustafa Kadhm et al. [15] proposed a model for forecasting diabetes by means ofKNN algorithm. They compared the KNN algorithm with the existing system, which used a Decision tree. By the End of the work, the proposed method is identified as the best, with an Accuracy of 98.7% compared to the existing system.

Pradeep Kandhasamy and Balamurali [16] compared various classifiers to predict diabetes. The classifiers they reached are J48, K-NN, RF and SVM. They used Dataset from the UCI machine learning repository.The comparison results concluded that J48 obtained better accuracy in the case of noisy data and RF, and KNN for k=100 obtained better accuracy in the case of preprocessed data.

Prema and Pushpalatha [17] mainly focused on evaluating the risk factors of GDM using data mining techniques. They implemented the K means clustering algorithm and J48, RF and NB classifiers on the Dataset. The Dataset is collected from the hospitals of Mysore. They focused on improving the accuracy of classifiers using wrapper feature subset selection. They compared the accuracy of classifiers for both balanced and unbalanced datasets. Among all the three classifiers, Random forest has obtained the best accuracy.

Renuka Devi and Maria Shyla [18] presented their work on analyzing data mining techniques using various tools. They considered previous papers that used the PIMA Indian diabetes dataset, containing 768 instances. They compared the results of algorithms performed by several authors. The analysis concluded that the modified J48 algorithm implemented using WEKA and MATLAB had obtained the best accuracy of 99.87% compared to other algorithms.

Rithesh [19] focused on the performance of SVM and KNN algorithms on various datasets. He proposed the SVM-KNN model, which implements SVM, k means clustering and KNN. He considered the diabetes dataset from the UCI ML repository. From the experimental results of the Diabetes dataset, he concluded SVM-KNN model had obtained better accuracy.

Suryakirani et al. [20] have analyzed four classification algorithms, J48, Random tree, Decision tree and NB classifier, for Diabetic Dataset. The diabetic Dataset consists of attributes like the number of pregnancies previously, plasma glucose concentration, age, skin thickness, BMI etc. The performance measures are computing time, correctly classified instances, kappa statistics, precision, Recall and F-Measure. Among all the algorithms, J48 showed better accuracy of 73.28%.

Using hybrid classifiers, Saradha and Sujatha [21] set out to create a system for predicting and diagnosing GDM. The SVM and J48 classifiers had their parameters tweaked to get the desired results. They used a data set obtained from various hospitals and clinical labs and approaches carried out using the WEKA tool. When evaluating the classifiers, modified J48 was found to have the highest accuracy, at 96.84%.

Data mining methods were used in the Dataset by SrideivanaiNagarajan et al. [22] to enhance the detection of gestational diabetes in pregnant patients. On the clinical Dataset, they used the ID3, NB, Random Forest, and C4.5 algorithms. The Dataset includes Information from 600 patients, all pregnant women over the age of 21. The best algorithm is determined by its accuracy and error rate, two performance metrics. According to the experimental findings, random forest is the best method, with an error rate of 0.000 and an accuracy of 93.8%.

Sumangali et al. [23] presented their work on the early detection of diabetes. They implemented CART, Random forest also a combination of both these algorithms. They applied these algorithms to the Dataset collected from the UCI machine learning (ML) repository. The experimental results concluded that combining CART and random forest obtained better accuracy.

Saba Bashir et al. [24] used multiple ensemble classification techniques on diabetes datasets. The base classifiers used are ID3, C4.5 and CART. They implemented ensemble techniques like Majority voting, Adaptive boosting, Bayesian boosting, Stacking and Bagging. They considered two diabetes datasets, PIMA and BioStat, from the UCI ML repository. They concluded that bagging shows better performance in the case of both datasets.

Shiva Shankar et al. [25] mainly focused on classifying breast cancer, whether benign or malignant. Their work used five ML algorithms to classify breast cancer data. Sathya and Rajesh [26] focused on the prediction of diabetes using the ID3 algorithm. They considered the UCI repository diabetes dataset containing 50 attributes. They implemented the ID3 algorithm in three ways without data cleaning and data cleaning by using unsupervised and supervised learning methods. The implementation is done using the WEKA tool. It was concluded that data cleaning with the supervised learning method had obtained better accuracy among these three ways.

According to Fu, H., Cheng et al. [27], Glaucoma is a sustained eye ailment which causes unrecoverable sight. They explored two novel detection techniques using deep learning methods. Method 1 is M-Net that unravels the optic disc and optic cup segmentation. M-Net has a multi-scale convolutional network to acquire discriminative representations and output segmentation probability maps. Then this result was used to predict Glaucoma. Method 2 is DENetwhich is an ensemble network. This network gives the glaucoma recognition result from the unsegmented images. They analyzed the two with other appropriate methods using glaucoma datasets.

Sanaa AbouElhamayed[28] mainly focused on the performance of CART, KNN and PCA algorithms on various datasets. He considered the PIMA Indian diabetes dataset along with some other datasets. The CART algorithm has obtained better accuracy of 100% for the Diabetes dataset. By viewing the datasets' results, PCA has achieved better performance results.

Shiva Shankar Reddy et al. [29] proposed a scheme for detecting Type-1 and Type-2 Diabetes. In their method, four data mining techniques are used for detecting diabetes utilizing a patient's medical record. The algorithms used are the NB, SVM, DT and Adaboost-M1. They compared these algorithms using a voting strategy to obtain the best scheme. They received 95 % accuracy when 10-fold cross-validation was used.

S.S. Reddy et al. [30] have researched GDM, and also worked on DM prediction in [31], multiple ailments in [32], correlated ailments in [33], predicted readmission patients are admitted or not in [34], conglomerative schemes in [35] and Diagnosis of Diabetes in [36].

3. System Architecture

In figure 2.1 the first step in the developed model is to perform data preprocessing on the Dataset. The Dataset after preprocessing is divided into training and test datasets. The k-fold cross-validation is applied to the training data with different values of k as 3, 5 and 10. The model is trained by using training data from the cross-validation technique.

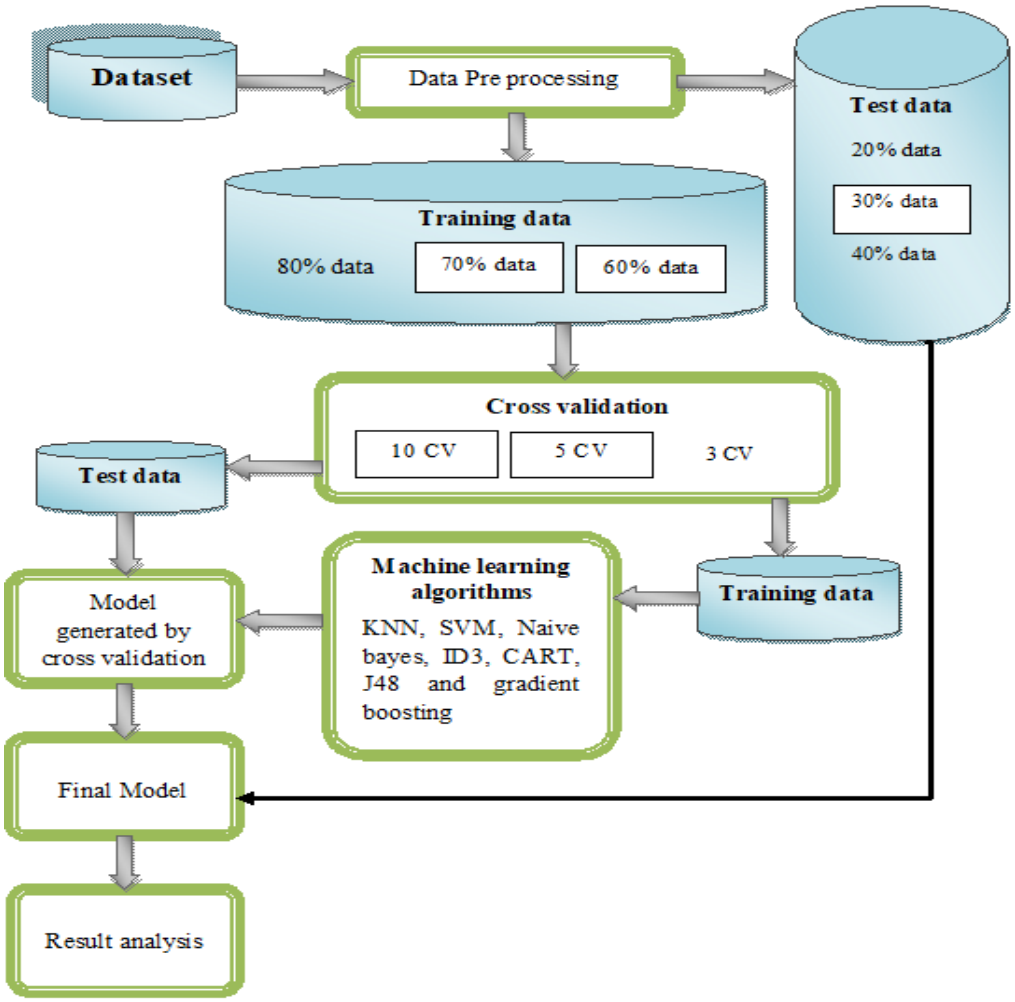

Figure 2.1. System architecture

The trained model is evaluated using test data of the cross-validation technique. Then the final model obtained after cross-validation is used to predict diabetes. The final model is assessed on the test data divided after preprocessing to get the results of each algorithm. In the result analysis, performance measures are compared for all the algorithms.

The implementation of algorithms is performed on Diabetes Dataset. Some attributes, like symptoms of diabetes, are added to the Pima Indian diabetes dataset [8]. The final Dataset contains 15 features with 768 samples of above 20 years aged women. It contains 14 predictors and 1 target variable.

A number of prior pregnancies, fasting glucose, 1-hour postprandial glucose, 2-hour postprandial glucose, 3-hour postprandial glucose, BP, and BMI are predictive factors. BMI, Insulin, Skin Thickness Age, polyuria, polydipsia, and diabetes family history function blur the eyesight. Class, a binary variable having the 0 and 1 values for negative and positive, is the variable of interest in the Dataset.

The Dataset undergoes preprocessing to prepare the data for analysis. Its primary goals are to normalize or scale the characteristics, distribute the Dataset for training and testing and fill in missing values. A percentage-based split separates the Dataset into training and testing data.

Data used for training is subjected to k-fold cross-validation. Data is partitioned into k equal subsets. Training data comprises k of the k-1 features here, whereas the test data comprises the remaining 1. The model is created using the cross-validation training data and then evaluated using the test data itself. By choosing portions for training and test data at random, this procedure is repeated k times. The ML methods used are described in depth in Sections 3.1 and 3.2.

3.1. Classification Algorithms Used

In this work, there are six prevalent techniques and one proposed method that are executed in R. The current algorithms are KNN, SVM, NB classifier, ID3, CART and J48. The proposed algorithm is Gradient boosting, which is an ensemble technique.

K-Nearest Neighbour (KNN)

KNN is a classification algorithm. It can identify the class to which a particular instance belongs. The Euclidean distance is calculated between a data point in the test dataset and all data points in the training dataset. Based on this distance, the output value will be predicted.

Algorithm 1: K-Nearest Neighbour (KNN)

Input: Each record of the data

Output: Predicted target class to which the test instance belongs based on k value.

Assumptions: k is the count of the nearest neighbours to be considered, d(x,y) is the Euclidean distance, x and y are data points in the training dataset and test dataset, respectively, xt and yt are data points in attribute t.

Step 1: Start

Step 2: Choose the value of k.

Step 3: For each data point in the test data, repeat step 4

Step 4: For all data points in the training data, repeat step 5

Step 5: Calculate the Euclidean distance between a data point in the test dataset for all data points in the Dataset considered for training.

$$d(x,y) = \sqrt{\sum_{t=1}^{n}(x_t - y_t)^2}$$

Step 6: End for in Step 4

Step 7: End for in Step 3

Step 8: Sort the data on Euclidean distance in increasing order.

Step 9: The result is the most frequent class from the first k rows.

Step 10: Stop

In step 2, fix the value of k, which is used in step 9. Step 3 indicates the data point in the test data. Step 4 suggests all data points in training data. In step 5, Euclidean distance is calculated between steps 3 and 4. The loops are ended in steps 6 and 7 for test and training data. All data points are sorted in step 8. In step 9, the first k data points are considered to get output in step 10.

Naive Bayes classifier

The NB classifier is the probabilistic algorithm. The higher value of probability gives a more accurate result. The test instance belongs to the higher probability class. The algorithm uses the following NB formula in the case of a single attribute

$$P\left(\frac{s}{t}\right) = \frac{P\left(\frac{t}{s}\right)P(s)}{P(t)} \qquad (1)$$

Where P(s/t) is a posterior probability, P(t/s) is likelihood and P(s) and P(t) are prior probabilities.

Algorithm 2: Naive Bayes classifier

Input: Each record of the data.

Output: Expected class to which the test instance belongs based on the posterior probability of course.

Assumptions: P(s/t) is the posterior probability of class c given features or attributes t. t represents multiple attributes, P(s) is the prior probability of class c, and P(t/s) is the likelihood probability for all attributes t and class c.

Step 1: Start

Step 2: Calculate the prior probability for class labels.

Step 3: Calculate the Likelihood probability for each class with each attribute.

Step 4: Multiply same class likelihood probability.

$$\mathbf{P}\left(\frac{\mathbf{t}}{\mathbf{s}}\right) = \mathbf{P}\left(\frac{\mathbf{t1}}{\mathbf{s}}\right)\mathbf{P}\left(\frac{\mathbf{t2}}{\mathbf{s}}\right) \ldots \ldots \mathbf{P}\left(\frac{\mathbf{tn}}{\mathbf{s}}\right)$$

Step 5: Calculate posterior probability using the Bayes formula

$$\mathbf{P}\left(\frac{\mathbf{s}}{\mathbf{t}}\right) = \mathbf{P}\left(\frac{\mathbf{t}}{\mathbf{s}}\right) * \mathbf{P}(\mathbf{s})$$

Step 6: Finally, the test instance belongs to the class with a higher posterior probability class.

Step 7: Stop

The initial probability is computed as given in step 2 for class labels used in step 5 to calculate posterior probability. In step 3, the likelihood probability is calculated. Step 4's likelihood probabilities of all the class labels are multiplied, which is used in the 5th step. In 5th step, the posterior probability is calculated based on the output predicted in step 6.

Support Vector Machine (SVM)

SVM is a supervised learning classification algorithm in which the Dataset is divided using the hyperplane. There are many possible hyperplanes to separate two classes of data points. The hyperplane with the maximum margin is to be selected. The prediction depends on which side of the support vector the test data point lies.

Algorithm 3: Support Vector Machine (SVM)

Input: Each data record

Output: Predicted target class to which the test instance belongs based on SVM.

Step 1: Start

Step 2: Distribute the Dataset into separate classes depending on the values.

Step 3: Construct hyperplanes to separate the Dataset into different classes.

Step 4: Select a hyperplane with the greatest margin.

Step 5: To which side of the region does the test data point belong in the output.

Step 6: Stop

In 2nd step, the data is categorized into separate groups. In 3rd step, all the possible hyperplanes between groups are constructed. In step 4, two hyperplanes with maximum margins are established and used to get the output in 5th step.

ID3

ID3 algorithm is a decision tree algorithm used for classification problems. ID3 uses information gain for the selection of splitting attributes. The information gain is calculated for each feature in the Dataset. The splitting attribute is selected based on the attribute with the highest information gain to construct a decision tree.

Algorithm 4: ID3

Input: Each data record.

Output: Expected test instance's target class depending on the decision tree generated using Information gain.
Assumptions: D is an attribute in the Dataset, g represents the class, Pt is the probability that a particular instance belongs to class t, A is a specific attribute, v means the different group of instances in attribute A, Dk is attributed to A with group v.
Step 1: Start

Step 2: For all attributes in the Dataset

a. Calculate the Entropy of the target attribute

$$E(D) = -\sum_{t=1}^{g} P_t \log_2(P_t)$$

b. Calculate Entropy for each attribute A.

$$E(D,A) = \sum_{k=1}^{V} \frac{|D_k|}{|D|} E(D_k)$$

c. Calculate Information gained for each attribute A

$$\text{Gain}(A) = E(D) - E(D,A)$$

Step 3: End for

Step 4: The highest information gain attribute is selected as the splitting attribute.
Step 5: The test data output is predicted based on leaf nodes.

In step 2a, calculate the Entropy of the target attribute. In step 2b, Calculate the Entropy for each attribute A. In step 2c, Calculate Information gain for each attribute 'A' using values obtained in steps 2a and 2b. In step 3, End for a loop. In step 4, the highest information gain attribute is selected to split the tree and used to predict output in step 5.

J48

J48 algorithm is a java implementation of the C4.5 and a descendant of ID3. It uses the gain ratio for the selection of splitting attributes. The gain ratio is defined as the normalization of information gain using the "split information" value. The splitting attribute is selected based on the attribute with the highest gain ratio to construct a decision tree.

Algorithm 5: J48

Input: : Each data record.

Output: Expected test instance'starget class depending on decision tree generated using Gain ratio.
Assumptions: D is the attribute in the Dataset, g represents the class, Pt is the probability that a particular instance belongs to class t, A is a specific attribute, v represents a different group of instances in attribute A, Dt is attributed to A with group v.

Step 1: Start

Step 2: For all attributes in the Dataset.

a. Calculate split Information for each attribute in the Dataset.

$$\text{SplitInfo}(D,A) = -\sum_{t=1}^{v} \frac{|D_t|}{|D|} \log_2\left(\frac{|D_t|}{|D|}\right)$$

b. Calculate the Gain ratio for each attribute.

$$\text{Gain ratio}(A) = \frac{\text{Gain}(A)}{\text{SplitInfo}(D,A)}$$

Step 3: End for

Step 4: The highest gain ratio attribute is selected as the splitting attribute.
Step 5: The test data output is predicted based on leaf nodes.
Step 6: Stop

In step 2a, calculate split Information for each attribute used in step 2b to calculate the Gain ratio. In step 2, the b gain ratio for each attribute is calculated using information gain in step 3, End for a loop. In step 4, the highest gain ratio attribute is selected to split the tree, which is used to predict the output in step 5.

Classification and Regression Tree (CART)

CART uses the Gini index to select splitting attributes. The Gini index measures impurity for the input data. It is calculated for each attribute. The splitting attribute is determined based on the attribute with the lowest Gini index to construct a decision tree. The higher value of the Gini index indicates homogeneity.

Algorithm 6: Classification and Regression Tree (CART)

Input: Each record of the data.

Output: Expected test instance's target class depending on the decision tree generated using the Gini index.
Assumptions: D is a set of training instances, g represents the class, Pt is the probability that instance belongs to class t in the target class, A is a specific attribute, D1 is attributed A with class 1, D2 is attributed A with class 2.
Step 1: Start

Step 2: For all attributes in the Dataset

a. Calculate Gini index of D

$$\text{Gini}(D) = 1 - \sum_{t=1}^{g} {P_t}^2$$

b. Calculate the weighted sum of the Gini indices of each attribute.

$$\text{Gini}(D,A) = \frac{|D_1|}{|D|}\,\text{Gini}(D_1) + \frac{|D_2|}{|D|}\,\text{Gini}(D_2)$$

Step 3: End for

Step 4: The splitting attribute is selected based on the lowest Gini index among all attributes.
Step 5: The test data output is predicted based on leaf nodes.
Step 6: Stop

In step, 2a Gini index of the attribute is calculated. In step 2b, the weighted sum of the Gini indices of each attribute A is calculated. In step 3, End for a loop. In step 4, the splitting

attribute is selected based on the Gini index values of all features. The output is predicted in step 5 based on leaf nodes.

Gradient boosting

Gradient boosting is an ensemble technique used for classification and regression problems. It trains the model in a gradual, additive and sequential way. It identifies weak learners using gradients in the loss function. Decision trees are used as vulnerable learners in Gradient boosting. It uses a gradient descent procedure while adding trees to reduce the loss. The output of the newly generated tree is added to the output of the previous trees. The loss function identifies how efficiently the predictive model classifies the data.

Algorithm 7: Gradient boosting

Input: Each data record, number of iterations and loss function L(y,Y).
Output: Gives the final model to predict the test data.

Assumptions: n is several instances in the training dataset, M is several iterations, $F_m(x)$ represents the model in iteration m, Fm-1(x) means the previous model of m, xt is instance t of attribute x, yt is instance t of attribute y, Y is a constant function.

Step 1: Start

Step 2: Initialize the model with a constant value

$$F_o(x) = \text{argmin}\gamma \sum_{t=1}^{m} L(y_t, \gamma)$$

Step 3: For m=1 to M

a. Compute the negative Gradient or pseudo residuals for each training instance t.

$$r_{tm} = -\left[\frac{\partial\left(L(y_t, F(x_t))\right)}{\partial F(x_t)}\right] \; ; \; Fm(x) = Fm\text{-}1(x)$$

b. Train the base learner hm(x) to pseudo residuals.

c. Compute multiplicative factor γm

$$\gamma_m = \text{argmin}\gamma \sum_{t=1}^{n} L(y_t, F_{m-1}(x_t) + \gamma h_m(x_t))$$

d. Update model Fm(x)

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

Step 4: End for

Step 5: Output Fm(x)

Step 6: Stop

In step 2, initialize the first model with a constant value. Step 3 is repeated for M iterations. In step 3a, calculate the pseudo residuals of one base learner. In steps 3, b, train the base learner using residuals in step 3a. In step 3c, calculate the multiplicative factor. In step 3d, update the model using Steps 3b and 3c. In step 4, End for a loop. In step 5, the final model is given as output.

3.2. Implementation Process

In implementing these algorithms, the basic packages required for methods of algorithms are loaded. These packages include 'caret', 'part', 'plyr', 'RWeka' and 'tidyverse'. The methods from packages are used to implement algorithms. The methods for KNN, SVM, NB classifier, ID3, CART, J48 and Gradient boosting are known, as SVM, nb, part, J48 and gbm. All the algorithms are trained using the train() function. The training dataset, the method used to train the algorithm, and cross-validation are given as parameters to train the () function.

The model will be trained and then evaluated with the test dataset using the predict() function. The trained model and target class of the test dataset are given as parameters to predict the () function. In chapter 4, the analysis of obtained results is provided.

4. Result Analysis

The performance measures like Accuracy, Kappa statistic, Sensitivity or Recall, Specificity, Precision, F-Measure and AUC are used to compare the results of all algorithms. For each algorithm, the performance measures are calculated from the confusion matrix. The confusion matrix contains True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN) are given in table 4.1. The confusion matrix of Gradient boosting obtained TP, TN, FP and FN values as 106, 37, 6 and 4.

Table 4.1. Confusion Matrix

		Predicted class	
		Positive	Negative
Actual class	Positive	TP	FP
	Negative	FN	TN

4.1. Performance Measures

The performance measures used for comparing algorithms are Accuracy, Kappa statistic, Sensitivity, Specificity, Precision, F-Measure and AUC for the ROC curve. All the performance measures are defined below. The performance measures for each algorithm are mentioned in section 4.2.

Accuracy

Accuracy is the ratio of correctly predicted cases to the total number of cases.

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

Accuracy of Gradient boosting
= (106 + 37) / (106 + 37 + 6 +4)=0.9346

Sensitivity or Recall

Sensitivity or Recall is the ratio of the records predicted as positive correctly to the total number of instances that are to be identified as positive.

$$\text{Sensitivity or Recall} = TP/((TP + FN))$$

Sensitivity of Gradient boosting
= 106 / (106 + 4)= 0.9636

Precision

Precision is the proportion of the number of instances predicted as positive correctly to the total number of cases indicated as positive.

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})}$$

Precision of gradient boosting = 106 / (106 + 6) = 0.9464

F-Measure

F-Measure is the harmonic mean between Precision and Recall.

$$\text{F} - \text{Measure} = \frac{(2 * \text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}$$

F – Measure of Gradient boosting
= (2 * 0.9464 * 0.9636) / (0.9464 + 0.9636) = 0.9549.

Kappa statistic

Kappa statistic is used for solving multi-class classification problems. It is also known as classification accuracy. The value of the kappa statistic ranges from -1 to +1. If it has the value 1, it is considered a good kappa statistic.

$$\text{Kappa} = \frac{(\text{Po} - \text{Pe})}{(1 - \text{Pe})}; \quad \text{Kappa} \leq 1$$

Here Po is observed Accuracy and Pe is expected accuracy. The Gradient boosting algorithm has obtained the value of kappa as 0.8359.

ROC – AUC curve

ROC- Receiver Operating Characteristic curve is a graph used to evaluate a classification model performance at all thresholds. This curve is plotted between two performance measures called True positive rate (TPR) or sensitivity and False positive rate (FPR). The false-positive rate is also defined as 1-specificity.

$$\text{True positive rate} = \frac{\text{TP}}{(\text{TP} + \text{FN})}$$

TPR of gradient boosting = 106 / (106 + 4) = 0.9636

$$\text{False positive rate} = \frac{\text{FP}}{(\text{FP} + \text{TN})}$$

FPR of gradient boosting = 6 / (6 + 37) = 0.1395

AUC- Area Under the ROC Curve is used to measure the full two-dimensional area under the ROC curve. The value of AUC lies between 0 and 1. If the value of AUC is 1 or nearer to 1 then that model prediction is correct.If the value of AUC is 0, then the model predictions are said to be incorrect.The Gradient boosting algorithms has obtained the area under the ROC curve (AUC) as 0.912.

4.2. Results obtained

Histogram of the attributes

Figure 4.1 contains a histogram of all the attributes in the Dataset. A multiple histogram graph with a histogram of each attribute is obtained for the Dataset.

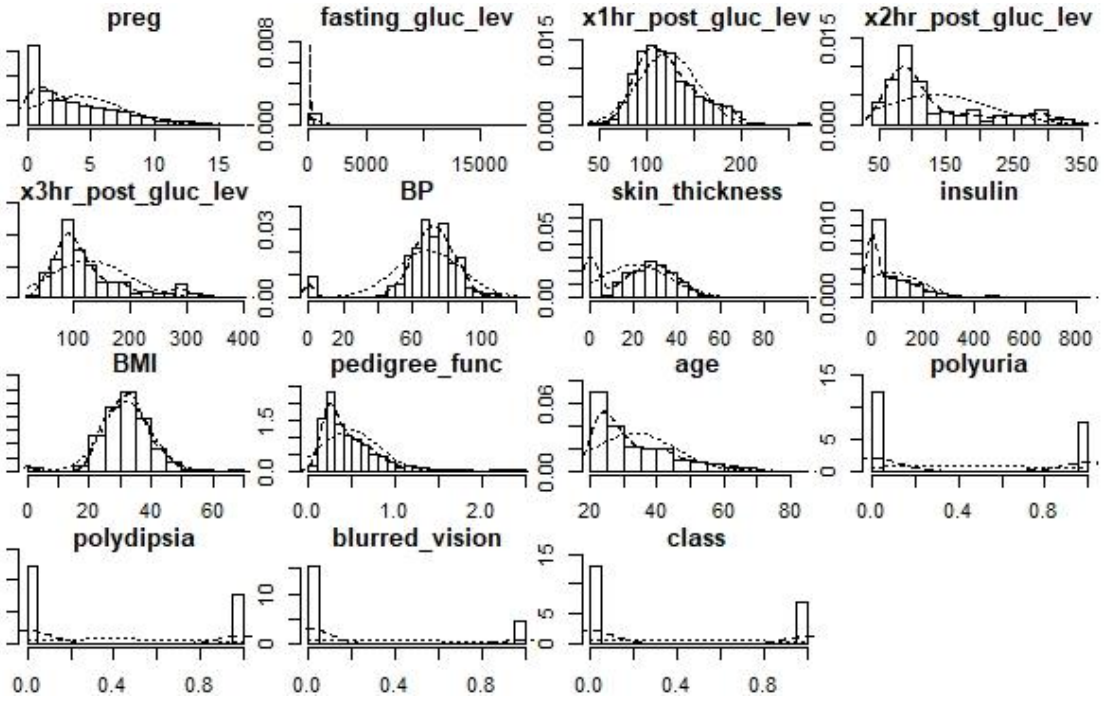

Figure 4.1. Histogram for all attributes in the Dataset

KNN using 10 CV results

Figure 4.2 shows the ROC curve obtained for 10 cross-validation KNN algorithms. The ROC curve is a graphical representation of the true positive rate against the false-positive rate obtained from the confusion matrix of KNN.
Figure 4.2 shows the ROC curve obtained for 10 cross-validation KNN algorithms. The ROC curve is a graphical representation of the TPR against the false-positive rate obtained from the confusion matrix of KNN.

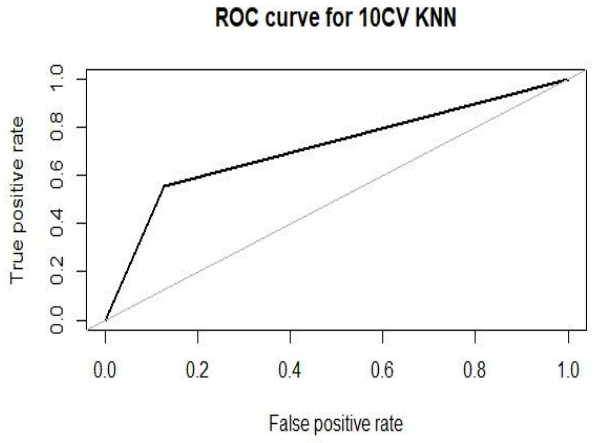

Figure 4.2. ROC curve for KNN using 10 CV

Table 4.2 shows performance measures of the KNN algorithm using ten cross-validations. These metrics are acquired from the confusion matrix of the KNN algorithm. The area Under ROC Curve is obtained from the ROC curve in figure 4.2.

Table 4.2. Performance measures of KNN using 10 CV

Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
78.43%	0.4467	0.8727	0.5581	0.8348	0.8533	0.715

SVM using 10 CV results

Figure 4.3 shows the ROC curve obtained for 10 cross-validation SVM algorithms. The ROC curve is a graphical representation of the TPR against the false-positive rate obtained from the confusion matrix of SVM.

Table 4.3 shows performance measures of the SVM algorithm using 10 cross-validations. These metrics are acquired from the confusion matrix of the SVM algorithm. The area Under ROC Curve is obtained from the ROC curve in figure 4.3.

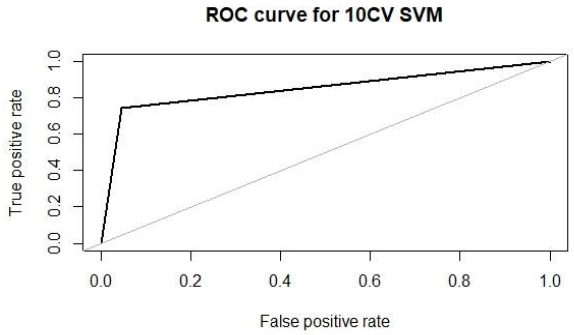

Figure 4.3. ROC curve for SVM using 10 CV
Table 4.3: Performance measures of SVM using 10 CV

Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
89.54%	0.7297	0.9545	0.7442	0.9052	0.9292	0.849

Naive Bayes classifier using 10 CV results

Figure 4.4 shows the ROC curve obtained for the 10 cross-validationsNB algorithms. The ROC curve represents the TPR against the false-positive rate obtained from the NB classifier's confusion matrix.

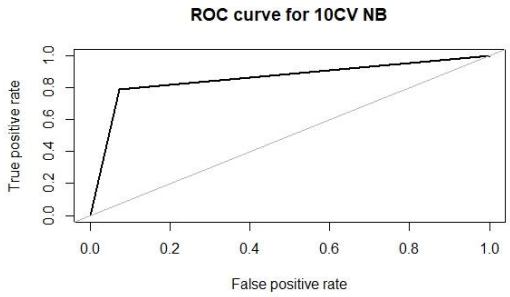

Figure 4.4. ROC curve for NB using 10 CV

Table 4.4 shows performance measures of the NB algorithm using 10 cross-validations. These metrics are acquired from the confusion matrix of the NB classifier. The area Under ROC Curve is obtained from the ROC curve in figure 4.4.

Table 4.4. Performance measures of Naive Bayes classifier using 10 CV

Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
88.89%	0.7231	0.9273	0.7907	0.9189	0.9230	0.859

ID3 using 10 CV results

Figure 4.5 shows the ROC curve obtained for 10 cross-validation ID3 algorithms. The ROC curve is a graphical representation of the TPR against the false-positive rate obtained from the confusion matrix of ID3.

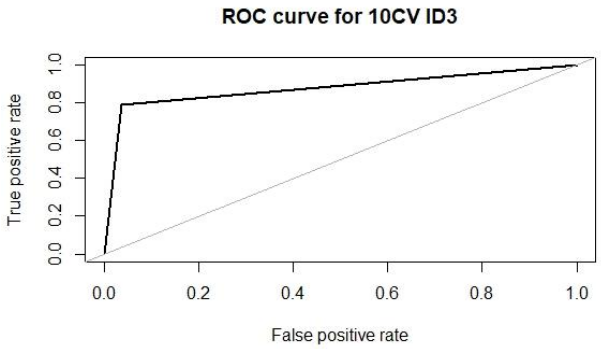

Figure 4.5. ROC curve for ID3 using 10 CV

Table 4.5 shows performance measures of the ID3 algorithm using 10 cross-validations. These performance measures are obtained from the confusion matrix of ID3. The area Under ROC Curve is acquired from the ROC curve in figure 4.5.

Table 4.5. Performance measures of ID3 using 10 CV

Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
91.5%	0.782	0.9636	0.7907	0.9217	0.9422	0.877

CART using 10 CV results

Figure 4.6 shows the ROC curve obtained for ten cross-validation CART algorithms. The ROC curve is a graphical representation of the TPR against the false-positive rate obtained from the confusion matrix of CART.

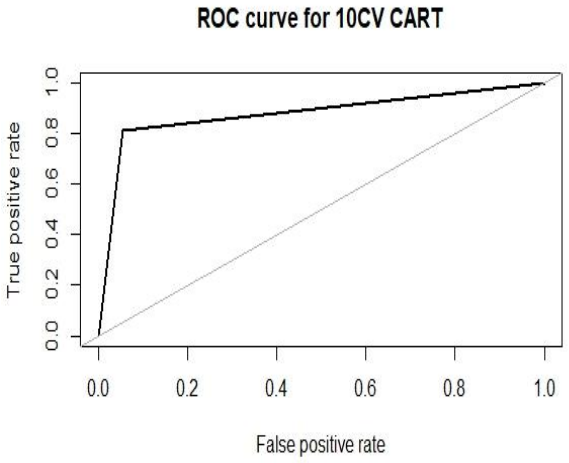

Figure 4.6. ROC curve for CART using 10 CV

Table 4.6 shows performance measures of the CART algorithm using 10 cross-validations. These performance measures are obtained from the confusion matrix of CART. The area Under ROC Curve is acquired from the ROC curve in figure 4.6.

Table 4.6. Performance measures of CART using 10 CV

Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
90.85%	0.7703	0.9455	0.8140	0.9286	0.9369	0.880

Table 4.7. Performance measures of J48 using 10 CV

Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
90.2%	0.7485	0.9545	0.7674	0.9130	0.9333	0.861

J48 using 10 CV results

Figure 4.7 shows the ROC curve obtained for 10 cross-validation J48 algorithms. The ROC curve is a graphical representation of the TPR against the false-positive rate obtained from the confusion matrix of J48.

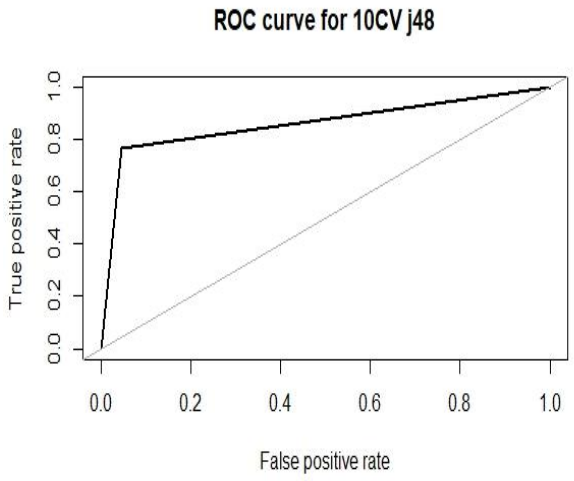

Figure 4.7. ROC curve for J48 using 10 CV

Table 4.7 shows performance measures of the J48 algorithm using 10 cross-validations. These metrics are acquired from the confusion matrix of J48. The area Under ROC Curve is acquired from the ROC curve in figure 4.7.

Gradient boosting using 10 CV results

Figure 4.8 shows the ROC curve obtained for 10 cross-validation gradient boosting algorithms. The ROC curve is a graphical representation of the TPR against the false-positive rate obtained from the confusion matrix of Gradient boosting.

Table 4.8 shows performance measures of the Gradient boosting algorithm using 10 cross-validations. These metrics are acquired from the confusion matrix of the Gradient boosting algorithm. The area Under ROC Curve is extracted from the ROC curve in figure 4.8.

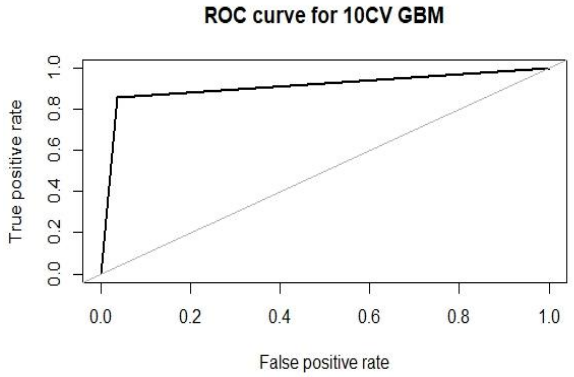

Figure 4.8. ROC curve for Gradient boosting using 10 CV

Table 4.8. Performance measures of angle boosting using 10 CV

Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC

93.46%	0.8359	0.9636	0.8605	0.9464	0.9549	0.912

4.2. Comparing Algorithms

Comparing results of algorithms using 10 CV

Table 4.9 shows all the performance measures obtained by implementing each algorithm. All the algorithms are implemented using 10 cross-validation techniques. In this table, a comparison of existing algorithms and the proposed algorithm is made. The proposed gradient boosting algorithm has obtained better performance measures.

Figure 4.9 illustrates the graph for comparing the accuracy of all the algorithms. Comparing the accuracy of 10 cross-validation techniques showed that Gradient boosting has obtained better accuracy.

By comparing the Accuracy, f-measure, and AUC of all the six existing algorithms using 10CV, ID3 showed the highest Accuracy, f-measure, and CART showed better AUC. Compare these results with the proposed algorithm results gradient boosting using 10CV. From the below graph in figure 4.10, it can be observed that Gradient boosting has obtained better AUC.

Table 4.9. Performance measures of all algorithms using 10 CV

Algorithm	Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
KNN	78.43%	0.4467	0.8727	0.5581	0.8348	0.8533	0.715
SVM	89.54%	0.7297	0.9545	0.7442	0.9052	0.9292	0.849
NB	88.89%	0.7231	0.9273	0.7907	0.9189	0.9230	0.859
ID3	91.5%	0.782	0.9636	0.7907	0.9217	0.9422	0.877
CART	90.85%	0.7703	0.9455	0.8140	0.9286	0.9369	0.880
J48	90.2%	0.7485	0.9545	0.7674	0.9130	0.9333	0.861
GBM	93.46%	0.8359	0.9636	0.8605	0.9464	0.9549	0.912

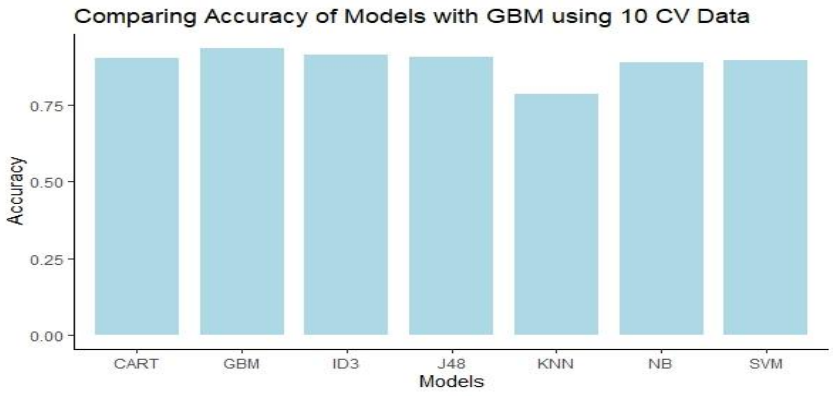

Figure 4.9. Comparing the accuracy of all algorithms using 10 CV

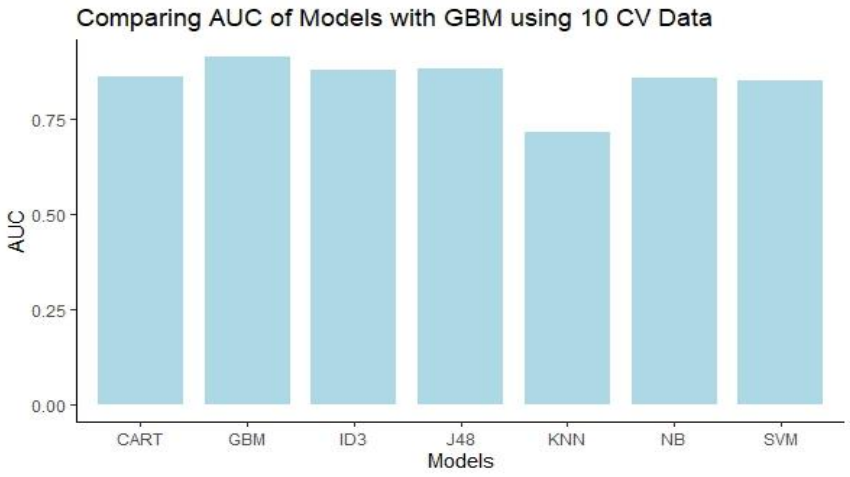

Figure 4.10. Comparing AUC of all algorithms using 10 CV

Comparing results of algorithms using 5 CV

Table 4.10 shows the values of all the performance measures obtained by implementing each algorithm. All the algorithms are implemented using 5 cross-validation techniques. In this table, a comparison of prevalent algorithms and the proposed algorithm is made. The proposed algorithm gradient boosting has obtained better performance measures.

Figure 4.11 illustrates the graph for comparing the accuracy of all the algorithms. Comparing the accuracies when 5 cross-validation technique is used shows that Gradient boosting has obtained better accuracy.

By comparing the Accuracy, f-measure, and AUC of all the six existing algorithms using 5CV, J48 showed the highest Accuracy, f-measure and AUC. Compare these results with the proposed algorithm gradient boosting using 5CV. From the below graph in figure 4.12, it can be observed that Gradient boosting has obtained better AUC.

Table 4.11. Performance measures of all algorithms using 5 CV

Algorithm	Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
KNN	77.83%	0.4743	0.8491	0.6197	0.8333	0.8411	0.703
SVM	90.43%	0.7649	0.9686	0.7606	0.9006	0.9333	0.865
NB	88.26%	0.726	0.9119	0.8169	0.9177	0.9148	0.864
ID3	91.3%	0.7914	0.9560	0.8169	0.9212	0.9382	0.886
CART	91.3%	0.793	0.9497	0.8310	0.9264	0.9378	0.890
J48	91.74%	0.8042	0.9497	0.8451	0.9321	0.9408	0.897
GBM	93.04%	0.8331	0.9686	0.8451	0.9333	0.9506	0.914

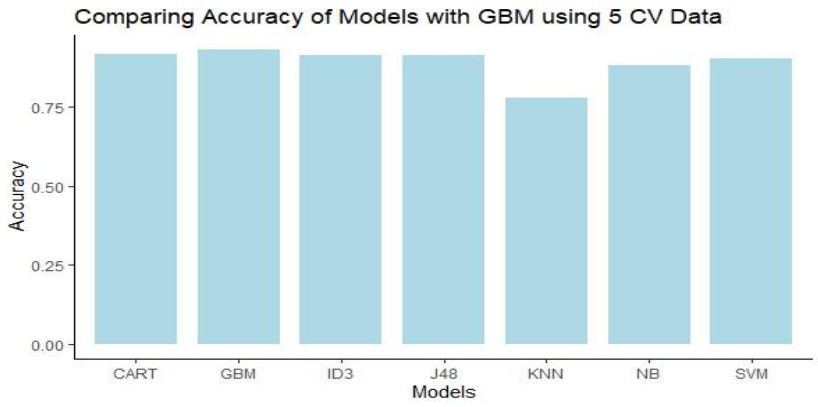

Figure 4.11. Comparing the accuracy of all algorithms using 5 CV

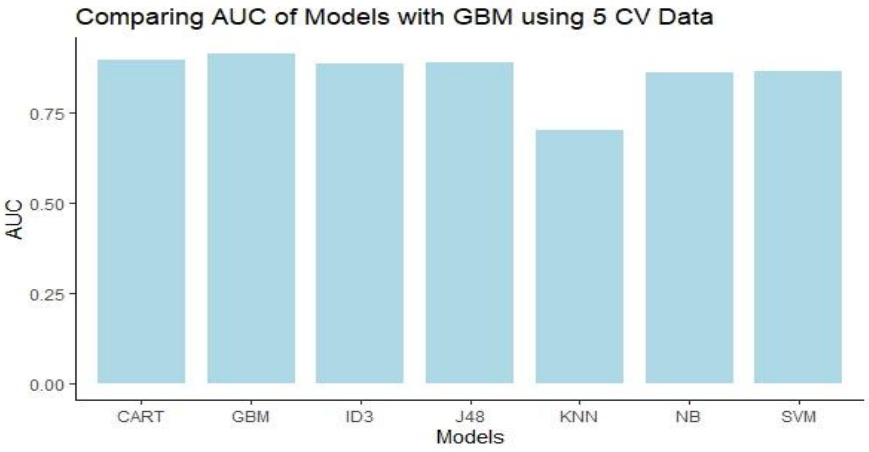

Figure 4.12. Comparing AUC of all algorithms using 5 CV

Comparing results of algorithms using 3 CV

Table 4.11 shows all the performance measures obtained by implementing each algorithm. All the algorithms are implemented using 3 cross-validation techniques. In this table, a comparison of existing algorithms and the proposed algorithm is made. The proposed algorithm gradient boosting has obtained better performance measures.

Figure 4.13 shows the plot for comparing the accuracy of all the algorithms. Comparing the accuracies when 3 cross-validation technique is used shows that Gradient boosting has obtained better accuracy.

Comparing the Accuracy, f-measure, and AUC of all the six existing algorithms using 3CV ID3 and CART showed the highest accuracy, and CART showed better AUC and f-measure. Compare these results with the proposed algorithm gradient boosting using 3CV. From the below graph in figure 4.14, it can be observed that Gradient boosting has obtained better AUC.

Table 4.11. Performance measures of all algorithms using 3 CV

Algorithm	Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
KNN	75.24%	0.4143	0.8381	0.5670	0.8073	0.8224	0.691
SVM	89.9%	0.7577	0.9571	0.7732	0.9013	0.9284	0.865
NB	88.27%	0.7302	0.9095	0.8247	0.9183	0.9138	0.809
ID3	92.51%	0.8242	0.9571	0.8557	0.9349	0.9458	0.906
CART	92.51%	0.8252	0.9524	0.8660	0.9390	0.9456	0.909
J48	92.18%	0.8171	0.9524	0.8557	0.9346	0.9433	0.907
GBM	94.14%	0.8605	0.9810	0.8557	0.9364	0.9581	0.926

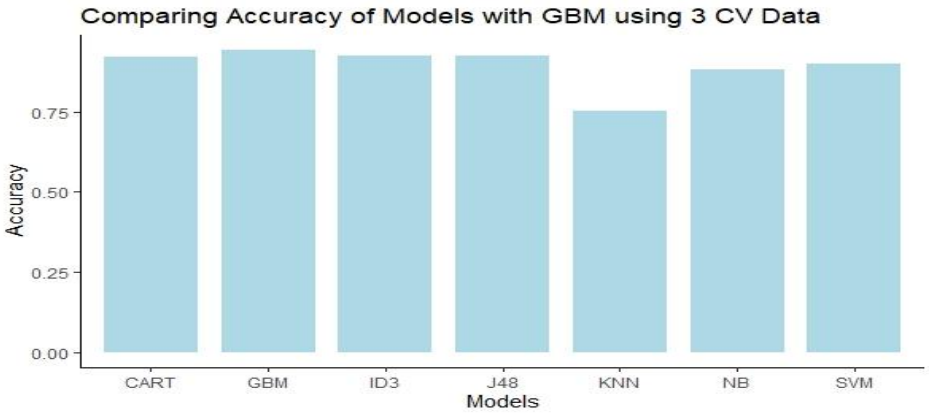

Figure 4.13. Comparing the accuracy of all algorithms using 3 CV

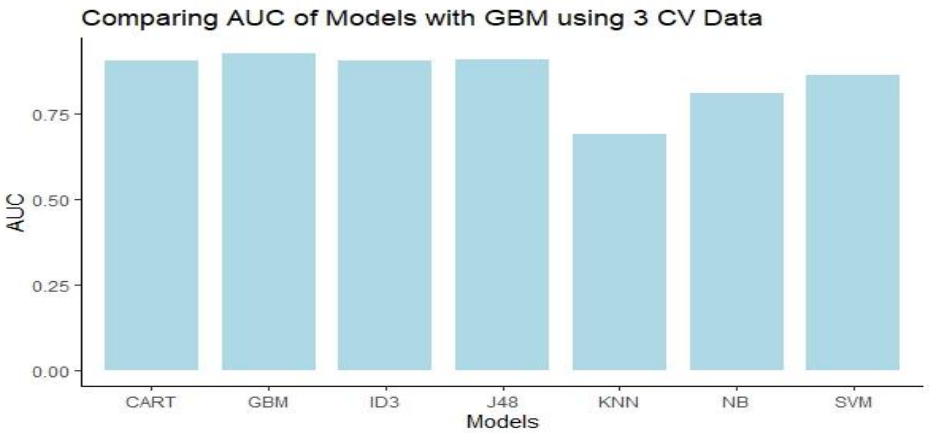

Figure 4.14. Comparing AUC of all algorithms using 3 CV

The ROC curve is shown only for algorithms implemented by applying 10-fold cross-validation. The performance measures for all algorithms with 3, 5 and 10 cross-validation are given above. By comparing the results of all algorithms of 10-fold cross-validation, it was observed that the Gradient boosting algorithm has the highest accuracy, kappa statistic, sensitivity, specificity, precision, f-measure and AUC with values 93.46%, 0.8359, 0.9636, 0.8605, 0.9404, 0.9549and 0.912 respectively. Similarly, from the results of 5 and 3 cross-validations,the gradient boosting algorithm obtains better performance measures.

4.3 Comparing ELM with GBM (Proposed Technique):

Extreme Learning Machine

ELM is a feed-forward neural network(NN)that could be used for classification. It contains one hidden layer, so a single hidden layer that feeds forward a neural network. The training of ELM is very fast compared to the artificial neural network. ELM has only a single input, hidden and output layers. It doesn't use a backpropagation algorithm like ANN. Instead, it utilizes an inverse matrix principle. It computes the output weight matrix using which the prediction is made. The stepwise algorithm for ELM is given below.

Algorithm 8: Extreme Learning Machine (ELM)

INPUT: Give the Dataset as input
OUTPUT: Predicted output values of input cases

ASSUMPTIONS: k is total records, $x_k$representsinput vector, h_t represents the output value for hidden neuron t and t = 1, 2,....,p, b_t is the bias for hidden neuron t, w_t represents weight vector for connections between the input layer and hidden layer neurons.t, β represents the weight vector connecting neurons of the hidden layer, and the output layer, g() is the activation function.

STEPS:

1. Start
2. Set values for the input layer neuron with the input instances.
3. Assign weights of input layer and biases for hidden layer neurons randomly.
4. Compute the output matrix for the hidden layer.

$$h_t = g(w_t * x_k + b_t), \text{where } k = 1,2,\ldots..,n \quad (7)$$

$$H = \begin{bmatrix} g(w_1x_1 + b_1) & \cdots & g(w_px_1 + b_p) \\ \vdots & \ddots & \vdots \\ g(w_1x_n + b_1) & \cdots & g(w_px_n + b_p) \end{bmatrix} \quad (8)$$

5. Obtain output layer weight matrix, the pseudo inverse of H. Here D is the matrix containing actual target values from the input instances.

$$\beta = (H * H^T)^{-1}H * D \quad (9)$$

$$\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} \quad (10)$$

6. Calculate output for the input instances.

$$T = H * \beta \quad (11)$$

$$T = \begin{bmatrix} t_1 \\ \vdots \\ t_p \end{bmatrix} \quad (12)$$

7. Return the predicted output values.
8. Stop

In 2nd step, the values for the input neuron are assigned based on input records. Assigning the input layer weights and the hidden layer neurons bias with random values to is illustrated in step 3. In step 4, the output matrix for the hidden layer is computed. The general output function given in Formula (7) is utilized to calculate each element of matrix H in formula (8). The activation function in this step is generally a sigmoid function. The matrix H from this step is utilized to obtain the weight matrix for the output layer in step 5. Here formula (9) is for getting the output weight matrix, and formula (10) shows matrix representation. The matrices H and β from steps 4 and 5 are utilized to compute the output values in step 6. The formula (11) calculates the output matrix with predicted target values. Formula (12) gives the matrix representation of T. Lastly, and the predicted values are given as output for the given input instances from this matrix T.

Results obtained for ELM

ELM with 3-fold cross-validation is implemented on the Dataset, and the results are obtained.

ELM is compared with GBM 3-fold cross-validation is implemented on the Dataset, and the results obtained are tabulated in Table 4.12.

Table 4.12. Performance measures of ELM and GBM using 3 CV

Algorithm	Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
ELM	90.02%	0.7593	0.9568	0.7909	0.9211	0.9363	0.885
GBM - Proposed	94.14%	0.8605	0.9810	0.8557	0.9364	0.9581	0.926

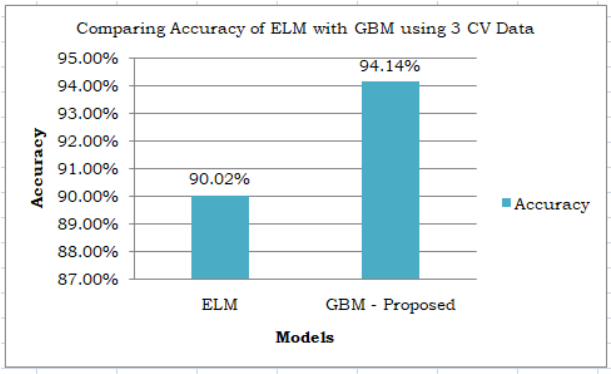

Figure 4.15. 3 CV Accuracy Comparison of ELM & GBM

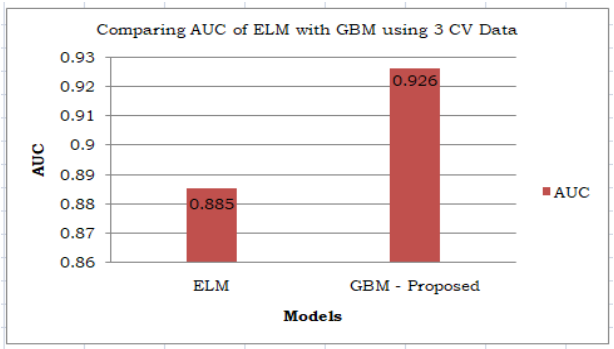

Figure 4.16. 3 CV AUCComparison of ELM & GBM

ELM is compared with GBM 5-fold cross-validation is implemented on the Dataset and the results obtained are tabulated in Table 4.13.

Table 4.13. Performance measures of ELM and GBM using 5 CV

Algorithm	Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
ELM	89.1%	0.7310	0.9417	0.7438	0.8819	0.9064	0.831
GBM - Proposed	93.04%	0.8331	0.9686	0.8451	0.9333	0.9506	0.914

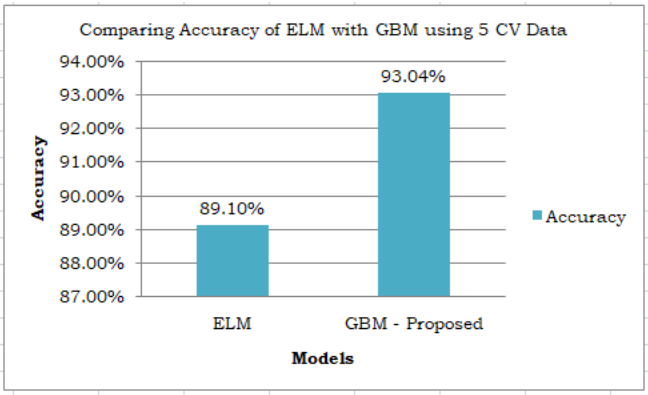

Figure 4.17. 5 CV Accuracy Comparison of ELM & GBM

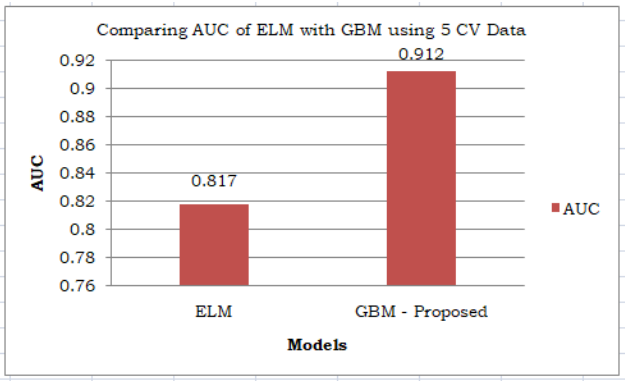

Figure 4.18. 5 CV AUCComparison of ELM & GBM

ELM is compared with GBM 10-fold cross-validation is implemented on the Dataset and the results obtained are tabulated in Table 4.14.

Table 4.14. Performance measures of ELM and GBM using 10 CV

Algorithm	Accuracy	Kappa	Sensitivity	Specificity	Precision	F-Measure	AUC
ELM	86.04%	0.7043	0.9108	0.7162	0.8716	0.8902	0.817
GBM - Proposed	93.46%	0.8359	0.9636	0.8605	0.9464	0.9549	0.912

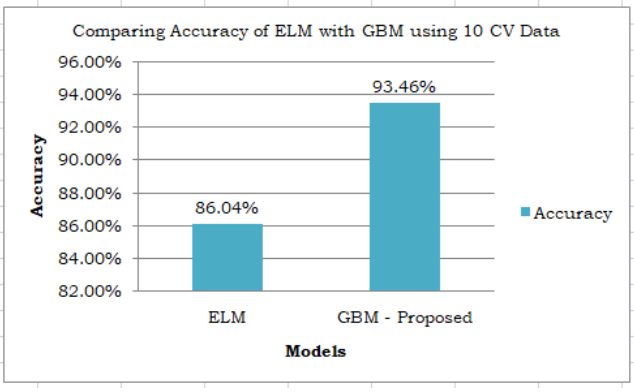

Figure 4.19. 10 CV Accuracy Comparison of ELM & GBM

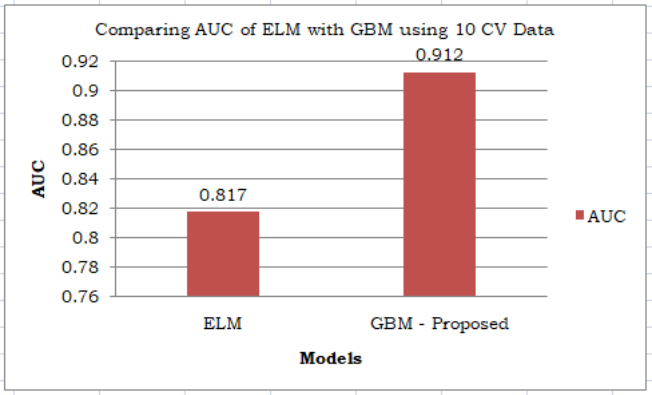

Figure 4.20. 10 CV AUCComparison of ELM & GBM

Tables 4.12, 4.13 and 4.14 show values of all the performance measures obtained by implementing each algorithm. All the algorithms are implemented using 3 cross-validation techniques, 5 cross-validation techniques, and 10 cross-validation techniques. In these tables, the comparison of existing algorithms and the proposed algorithm is made. The proposed algorithm gradient boosting has obtained better performance measures.

Figures 4.15, 4.17 and 4.19 show the plot for comparing the Accuracy of ELM with GBM. Figures 4.16, 4.18 and 4.20 show a plot for comparing the AUC of ELM with GBM. By comparing the accuracies when 3 cross-validations, 5 cross-validation techniques and 10 cross-validation techniques, it was observed that Gradient boosting had obtained better accuracy.

The accuracy, kappa statistic, sensitivity, specificity, precision, and f-measure are obtained using a function in the'caret' package called confusion matrix (). The ROC curve and AUC for true and false positive rates are obtained using the roc. curve() function in the'ROSE' package. In chapter 5, the conclusion is provided

5. Conclusion

A model to predict gestational diabetes in pregnant women is developed in this work. Several existing classification algorithms were implemented in R programming, like the NB classifier, SVM, KNN, ID3, CART and J48. The proposed algorithms Gradient boosting machine (GBM) and ELMarewere also implemented. K-fold cross-validation is applied with different values of k as 3, 5 and 10. Comparing the advanced ML technique ELM with the Gradient boosting machine proves that the proposed GBM obtained better results. Hence, it is concluded that GBM works better than other advanced techniques for the considered Dataset for gestational diabetes. The results showed that the Gradient boosting algorithm performs better than different algorithms. The Gradient boosting algorithm obtained better values of Accuracy and AUC of 94.14% and 0.926, respectively, when 3 cross-validations were applied. In the future, even better results can be obtained by using neural networks or deep neural networks.

References

[1] Melissa CS. Gestational Diabetes Signs, Symptoms, Test, Treatment, Complications, and Diet [online]. Medicine Net; [cited 2020 nov 22]. Available from: https://www.medicinenet.com/gestational_diabetes/article.htm

[2] Jenna F. What are the symptoms of gestational diabetes? [online]. Medical News Today; [cited 2020 nov 22]. Available from: https://www.medicalnewstoday.com/articles/325177.

[3] Mayo Clinic Staff. Gestational Diabetes [online]. Mayoclinic; [cited 2020 nov 22]. Available from: https://www.mayoclinic.org/diseases-conditions/gestational-diabetes/symptoms-causes/syc-20355339.

[4] Glucose screening tests during pregnancy [online]. Medline Plus; [cited 2020 nov 22]. Available from: https://medlineplus.gov/ency/article/007562.htm.

[5] Rohit G. 7 Types of Classification Algorithms [online]. Analytics India Magazine; [cited 2020 nov 22]. Available from: https://analyticsindiamag.com/7-types-classification-algorithms/.

[6] AmitabhaDey. Machine Learning (ML) - Data Pre processing[online]. , Data Driven Investor; [cited 2020 nov 22]. Available from: https://medium.com/datadriveninvestor/data-preprocessing-for-machine-learning-188e9eef1d2c.

[7] RenuKhandelwal. K fold and other cross-validation techniques[online]. , Data Driven Investor; [cited 2020 nov 22]. Available from: https://medium.com/datadriveninvestor/k-fold-and-other-cross-validation-techniques-6c03a2563f1e.

[8] Pima Indians Diabetes Database [online]. data.world; [cited 2020 nov 22]. Available from: https://data.world/data-society/pima-indians-diabetes-database.

[9] Appaji SV, Shankar RS, Murthy KV, Rao CS. Cardiotocography Class Status Prediction Using Machine Learning Techniques. Indian Journal of Public Health Research & Development. 2019;10(8):651-7.

[10] Vasamsetty CS, Peri SR, Rao AA, Srinivas K, Someswararao C. Gene Expression Analysis31for Type-2 Diabetes Mellitus-A Case Study on Healthy vs Diabetes with ParentalHistory. International Journal of Engineering and Technology. 2011 Jun 1;3(3):310-314.

[11] Vasamsetty CS, Peri SR, Rao AA, Srinivas K, Someswararao c. Gene Expression Analysis for Type-2 Diabetes Mellitus--A Study on Diabetes with and without Parental History. Journal of Theoretical & Applied Information Technology. 2011 May 15;27(1):43-53.

[12] Geetha VR, Jayaveeran, N. Comparative analysis of gestational diabetes using data mining techniques. Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol. 2018; 3(8): 2456-3307

[13] Mala SJ. A Hybrid Apporach of Classification Techniques for Predicting Diabetes using Feature Selection. International Journal of Trend in Scientific Research and Development.2019;3(5):2506-2510.

[14] Koklu M, Unal Y. Analysis of a population of diabetic patients databases with classifiers. International Journal of Biomedical and Biological Engineering. 2013 Aug 22;7(8):481-3.

[15] Kadhm MS, Ghindawi IW, Mhawi DE. An accurate diabetes prediction system based on K-means clustering and proposed

classification approach. International Journal of Applied Engineering Research. 2018; 13(6):4038-41.
[16] Kandhasamy JP, Balamurali SJ. Performance analysis of classifier models to predict diabetes mellitus. Procedia Computer Science. 2015 Jan 1;47:45-51.
[17] Prema NS, Pushpalatha MP. Evaluation of Risk Factors of Gestational Diabetes Mellitus (GDM) using Data Mining. International Journal of Engineering and Advanced Technology. 2019; 8(6): 695-698.
[18] Renuka DM, Shyla JM. Analysis of Various Data Mining Techniques to Predict Diabetes Mellitus. Int. J. Appl. Eng. Res. 2016;11(1):727-730.
[19] Rithesh RN. SVM-KNN: a novel approach to classification based on svm and knn. International Research Journal of Computer Science. 2017;4(8): 43-49.
[20] Suryakirani, RS and Porkodi, R. Comparative Study and Analysis of Classification Algorithms in Data Mining Using Diabetic Dataset. IJSRST. 2018; 4(2):299-304.
[21] Saradha S, Sujatha P. Prediction of gestational diabetes diagnosis using SVM and J48 classifier model. International Journal of Engineering & Technology. 2018;7(2.21):323-326.
[22] Srideivanai N, Chandrasekaran RM. and Ramasubramanian P. Data Mining Techniques for Performance Evaluation of Diagnosis in Gestational Diabetes. International Journal of Current Research and Academic Review. 2014; 2(10):91-98
[23] Sumangali K., Geetika, BSR. andHarshithaA.Author AA. A Classifier Based Approach for Early Detection of Diabetes Mellitus. In: Proceedings of the International Conf. on Control, Instrumentation, Communication and Computational Technologies; 2016; Kumaracoil, India. IEEE; 2016. p. 389-392.
[24] Bashir S, Qamar U, Khan FH, Javed MY. An efficient rule-based classification of diabetes using ID3, C4. 5, & CART ensembles. In: Proceedings of the 12th International Conference on Frontiers of Information Technology; 2014 Dec 17-14; Islamabad, Pakistan. IEEE; 2014. p. 226-231.
[25] Shankar RS, Gupta VM, Murthy KV, Rao CS. Breast cancer Data classification Using Machine Learning Mechanisms. Indian Journal of Public Health Research & Development. 2019;10(5):214-220.
[26] Sathya S. and Rajesh A.An Effective Prediction of Diabetics using ID3 Classification Algorithm. Middle-East Journal of Scientific Research. 2016; 24: 207-211
[27] Fu H, Cheng J, Xu Y and Liu J. Glaucoma detection based on deep learning network in fundus image. In: Deep learning and convolutional neural networks for medical imaging and clinical informatics;. Springer, Cham; 2019. p. 119-137.
[28] Sanaa AEl. Classifying Datasets using some Different Classification Methods. International Journal of Engineering and Technical Research. 2016),.5(2):148-154.
[29] Reddy SS, Rajender R, Sethi N. A data mining scheme for detection and classification of diabetes mellitus using voting expert strategy. International Journal of Knowledge-Based and Intelligent Engineering Systems. 2019 Jan 1;23(2):103-108.
[30] Reddy SS, Sethi N, Rajender R. Rigorous assessment of data mining algorithms in gestational diabetes mellitus prediction. International Journal of Knowledge-based and Intelligent Engineering Systems. 2021 Jan 1;25(4):369-83.
[31] Reddy SS, Sethi N, Rajender R. A Comprehensive Analysis of Machine Learning Techniques for Incessant Prediction of Diabetes Mellitus. International Journal of Grid and Distributed Computing. 2020;13(1):1-22.
[32] Reddy SS, Sethi N, Rajender R. Mining of multiple ailments correlated to diabetes mellitus. Evolutionary Intelligence. 2021 Jun;14(2):733-40.
[33] Reddy SS, Sethi N, Rajender R. A review of data mining schemes for prediction of diabetes mellitus and correlated ailments. In2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA) 2019 Sep 19 (pp. 1-5). IEEE.
[34] Reddy SS, Sethi N, Rajender R. Evaluation of deep belief network to predict hospital readmission of diabetic patients. In2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA) 2020 Jul 15 (pp. 5-9). IEEE.
[35] Reddy SS, Sethi N, Rajender R. Safe Prediction of Diabetes Mellitus Using Weighted Conglomeration of Mining Schemes. In2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA) 2020 Nov 5 (pp. 1213-1220). IEEE.
[36] Reddy SS, Sethi N, Rajender R, Vetukuri V. Non-invasive Diagnosis of Diabetes Using Chaotic Features and Genetic Learning. InInternational Conference on Image Processing and Capsule Networks 2022 (pp. 161-170). Springer, Cham.