Weighted Rough Classification for Imbalanced Gene Expression Data

E. N. Sathishkumar
Department of Computer Science
Periyar University
Salem, India
en.sathishkumar@yahoo.in

K. Thangavel
Department of Computer Science
Periyar University
Salem, India
drktvelu@yahoo.com

P. S. Raja
Department of Computer Science
Periyar University
Salem, India
psraja5@gmail.com

***Abstract*— Classification is an important task in data mining and its applications. In this paper, a novel method Weighted Rough Classification (WRC) using Neural Network is proposed to handle inconsistent and uncertain dataset. This method exploits the framework of rough set theory. The rough set is defined as a pair of lower approximation and upper approximation. The difference between upper and lower approximation is defined as boundary. In the traditional rough set, all samples have equal weight without considering the distribution of samples. Here, we apply Class equal Sample Weighting (CSW) to build a weighted information table. By using this method, samples belonging to majority class have smaller weight while samples in the minority class have larger weight. Based on the weighted information system we perform the 'inference decision making'. The weighted values are to be discretized since rough set based classification is proposed. The boundary values of weighted information system are considered as uncertain values, and inference decision making is done based on the similarity measure. The similarity measure is evaluated for elements of boundary with the centroid of each class lower approximation, and the decision value is updated according to the closest centroid. Experimental analysis shows that the Novel Weighted Rough Classification algorithm is effective and suitable for evaluating the classification. Further, experimental studies shows that WRC based technique outperformed, compared with the existing techniques.**

Keywords—Rough Set, Weighted Rough Classification, Classification, Gene Expression, Neural Network.

1. INTRODUCTION

Classification techniques are one of the important components of machine learning in order to extract patterns from huge data that could be used for prediction. Classification is a method of mapping data records into one of several predefined classes. It is the process of finding a set of models that describe and distinguish data classes and concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown (Han & Kamber, 2001). A classifier is built by following two steps namely training and testing. In training phase, classification models are built. The individual objects or examples are referred collectively as training dataset. Before building the model, this training set should be classified i.e., to attach a class label to each object or example. In testing phase, the model built in the previous step is used for prediction.

For microarray data, the applications of classification include diagnosing cancer type, given the expression pattern from a tumor sample, predicting the biological function of genes based on their expression patterns, and decides which class a presented sample belongs to. This paper presents a brief overview of several classification methods, with a particular focus on rough set approaches. The weighted Rough Classification using Neural Network has been proposed to classify the gene expression data. The performance of eight classification algorithms such as Naïve Bayes, JRip, J48, Random Forest, Decision Table, K-Star from WEKA (Kavitha & Mahalekshmi, 2015), Backpropagation Neural Network (BPN), and proposed Weighted Rough Classification have been compared. A brief explanation of each of the classification techniques are applied in this paper is presented in following sections.

Section 2 provides brief introduction about Rough Set Theory and its techniques. Section 3 Explains the Classification Models. Section 4 explains proposed novel Weighted Rough Classification Model. Section 5 details the performance evaluation measures of classification methods. Section 6 provides the experimental results and analysis. Finally, the Conclusion of this paper is presented in section 7.

2. ROUGH SET THEORY

Rough Set Theory approach involves the concept of indiscernibility. Let IS = (U,A,C,D) be a decision system data, where U is a non-empty finite set called the universe, A is a set of features, C and D are subsets of A, named the conditional and decisional attributes subsets respectively. The elements of U are called objects, cases, instances or observations. Attributes are interpreted as features, variables or characteristics conditions. Given a feature a, such that: a: U → Va for a ∈ A, Va is called the value set of a. Let a ∈ A, P ⊆ A, the indiscernibility relation IND(P), is defined as follows:

$$IND(P)=\{(x, y) \in U\times U : \text{ for all } a \in P, a(x) = a(y)\} \quad (1)$$

The partition generated by IND(P) is denoted as U/IND(P) or abbreviated to U/P and is calculated as follows

$$U/IND(P) = \otimes\{a \in P|U/IND(\{a\})\} \quad (2)$$

where $A \otimes B = \{X \cap Y \,|\forall X \in A, \forall Y \in B, X \cap Y \neq \emptyset\}$ where A and B are families of sets. If (x, y) ∈ IND(P), then x and y are indiscernible by attributes from P. The equivalence classes of the P-indiscernibility relation are denoted by [x]P.

2.1 Lower approximation of a subset

Let R ⊆ C and X ⊆ U, the R-lower approximation set of X, is the set of all elements of U which can be with certainty classified as elements of X.

$$RX = \cup \{ Y \in U / R : Y \subseteq X\} \quad (3)$$

According to this definition, we can see that R-Lower approximation is a subset of X, thus RX ⊆ X.

2.2 Upper approximation of a subset

The R-upper approximation set of X is the set of all element of U, which can possibly belong to the subset of interest X.

$$\bar{R}X = \cup \{ Y \in U / R : Y \cap X \neq \varphi \} \quad (4)$$

2.3 Boundary Region

It is the collection of elementary sets defined by:

$$BND(X) = \bar{R}X - RX \quad (5)$$

These sets are included in R-Upper but not in R-Lower approximations. A subset defined through its lower and upper approximations is called a Rough set . That is, when the boundary region is a non -empty set ($R\bar{X} \neq RX$).

3. CLASSIFICATION TECHNIQUES

3.1. Naive Bayes Classifier

Naïve Bayes algorithm is found in the WEKA classifiers under Bayes method. The Naive Bayesian classifier is based on Bayes theorem with independence assumptions between predictors. A Naive Bayesian model is easy to build and can be used for very large datasets. It can handle an arbitrary number of independent variables whether continuous or categorical. This algorithm makes prediction using Bayes theorem which incorporates evidence or prior knowledge in its prediction (Fauset, 1994) (Forbes, 1995). Naive Bayesian classifier often performs well. The posterior probability, P(c|x) is calculated from P(c), P(x), and P(x|c). The effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence. Formula for calculating posterior probability is given in equation (6).

$$P(c|x) = \frac{P(X|C)\,P(c)}{P(x)} \quad (6)$$

- P(c|x) is the posterior probability of class given predictor.
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predict or given class.
- P(x) is the prior probability of predictor.

One way of classification is by determining the posterior probability for each class and assigning c to the class with the highest probability.

In simple terms, a Naive Bayes classifier assumes that the presence or absence of a particular feature of a class is unrelated to the presence or absence of any other features, given the class variable. Depending on the precise nature of the probability model, Naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for Naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods. In spite of their naive design and apparently over-simplified assumptions, Naive Bayes classifiers have worked quite well in many complex real-world situations. Analysis of the Bayesian classification problem has shown that there are some theoretical reasons for the apparently unreasonable efficacy of Naive Bayes classifiers. An advantage of the naive Bayes classifier is that it only requires a small amount of training data to estimate the parameters necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

3.2. JRip Classifier (JRipper)

JRip algorithm is available in the WEKA classifiers system under Rules method. JRip and this implements a propositional rule learner. William W. JRip proposed a Repeated Incremental Pruning to Produce Error Reduction (RIPPER) (Witten & Frank, 2005) which is one of the basic and most popular algorithms. It is an inference and rules-based learner that can be used to classify elements with propositional rules (Devasena, Sumathi, Gomathi, & Hemalatha, 2011) (Thangaraj & Vijayalakshmi, 2013). The Ripper algorithm is a direct method used to extracts the rules directly from the data. Classes are examined in increasing size and an initial set of rules for the class is generated using incremental reduced-error pruning. The algorithm progresses through four phases: i) Growth ii) Pruning iii) Optimization iv) Selection. In this study, we evaluated RIPPER through JRip, an implementation of RIPPER in WEKA with the parameters initialization, such as folds = 10; minNo = 2; optimizations = 2; seed = 1; usePruning = true.\

3.3. J48 Classifier

The J48 algorithm is implemented in WEKA which is based on C4.5 decision tree learner. Pruning takes place by replacing internal node with a leaf node by using Hunt's algorithm (Kavitha & Mahalekshmi, 2015) (Salzberg, 1994). This algorithm uses a greedy technique to induce decision trees for classification and uses reduced-error pruning. Decision tree is a predictive machine learning model that decides the target value of a new sample based on various attribute values of the available data. The internal nodes of a decision tree denote the different attributes, the branches between the nodes gives the possible values that these attributes can have in the observed samples, while the terminal nodes gives the final value of the dependent variable. The attribute that is to be predicted is known as the dependent variable, since its value depends upon, or is decided by, the values of all the other attributes. The other attributes, which help in predicting the value of the dependent variable, are known as the independent variables in the dataset. In order to classify a new item, a decision tree is created based on the attribute values of the available training data. Whenever it encounters a training set of items, it identifies the attribute that discriminates the various instances most clearly. This feature that gives the most about the data instances can be classified as the best is said to have the highest information gain. Among the possible values of this feature, if there is any value for which there is no ambiguity, that is, for which the data instances falling within its category have the same value for the target variable, then that branch is

terminated and the target value that we have obtained is assigned to it. This is continued in this manner until we either get a clear decision of what combination of attributes gives us a particular target value, or we run out of attributes. In the event of running out of attributes, or if an unambiguous result is obtained from the available information, then this branch is assigned a target value that the majority of the items under this branch possess.

3.4. Random Forest Classifier

Random Forest (RF) algorithm is available in WEKA classifier under Trees method. Random Forest classifiers are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest, and it was developed by Breiman et al. (Breiman, 2001), which works on the concept of generating multiple Random Trees, with the help of bootstrap of training dataset trees, one has to set random forest before execution of an algorithm like, number of trees in the forests, depth of the tree, number of samples for bagging, number of features for splitting node. Random Forests gives many classification trees without pruning. Each classification tree gives a certain number of votes for each class. Among all the trees, the algorithm chooses the classification with the most number of votes. Random forest runs efficiently on large datasets but is comparatively slower than other algorithms. It can estimate missing values and hence it is suitable for handling datasets with large number of missing values.

The main advantage of using Random Forest is because of its randomness, so that it doesn't depend of the data and it is good with dealing with the outliers. Random Forest is good at dealing with high dimensional data, and performance and the accuracy results of random forests are very much promising and comparable with some of the state of the art techniques (Breiman, 2001). One of the best things that random forest adds is the randomness on to the data while it classifies, and due to randomness each tree would be highly uncorrelated with the other random tree. If trees would be correlated then it would not give the proper results of accuracy, but due to highly uncorrelated tree, when we combine those trees using bagging approach, results would be much improved. When compared to those, which are highly correlated and generating the most similar results all the time that shows a classifier Random Forests can provide a good accuracy, and can efficiently handle outliers.

3.5. Decision Table Classifier

The decision table classifier algorithm is available in WEKA classifier under Rules method. A decision table specifies only the logical rules. It is used to find out the decision quality. It evaluates feature subsets using best-first search and can use cross-validation for evaluation. There is a set of methods such as Best First, Rank Search, and Genetic Search; those can be used in the search phase. Decision Table algorithm builds and using a simple decision table majority classifier. It summarizes the dataset with a decision table which contains the same number of attributes as the original dataset. Then, a new data item is assigned a category by finding the line in the decision table that matches the non-class values of the data item. Decision Table employs the wrapper method to find a good subset of attributes for inclusion in the table. By eliminating attributes that contribute little or nothing to a model of the dataset, the algorithm reduces the likelihood of over-fitting and creates a smaller and condensed decision table. The output will show a decision on a number of attributes for each instance. The number and specific types of attributes can vary to suit the needs of the task. Two variants of decision table classifiers are,

- DTMaj (Decision Table Majority)
- DTLoc(Decision Table Local)

Decision Table Majority returns the majority of the training set if the decision table cell matching the new instance is empty, that is, it does not contain any training instances. Decision Table Local is a new variant that searches for a decision table entry with fewer matching attributes (larger cells) if the matching cell is empty. Hence this variant returns an answer from the native region (Tiwary, 2014) (Sharma & Jain, 2013).

3.6. K-Star Classifier

The K-Star algorithm is available in WEKA classifier under Lazy learning classification. It can be defined as a method of cluster analysis which mainly aims at the partition of 'n' observations into 'K' clusters in which each observation belongs to a cluster with the nearest mean. One can describe K-Star algorithm as an instance based learner which uses entropy as a distance measure. The benefits are that it provides a consistent approach to handling of real valued attributes, symbolic attributes and missing values (Cleary & Trigg, 1995) (Painuli, Elangovan, & Sugumaran, 2014).

K-Star is a simple, instance based classifier, similar to K-Nearest Neighbor (K-NN). New data instances, x, are assigned to the class that occurs most frequently amongst the K-nearest data points, yj, where j = 1, 2,…, K. Entropic distance is then used to retrieve the most similar instances from the data set. By means of entropic distance as a metric has a number of benefits including handling of real valued attributes and missing values. The K-Star function can be calculated as:

$$K^*(y_i, x) = -\ln P^*(y_i, x) \tag{7}$$

where P* is the probability of all transformational paths from instance x to yi. It can be useful to understand this as the probability that x will arrive at yi via a random walk in feature space. It will perform optimization over the percent blending ratio parameter which is analogous to K-NN 'sphere of influence', prior to assessment with other Machine Learning methods. Space required for the storage is very large as compared to other algorithms. Mostly noisy training data increases the case support unnecessary. It is usually slower in evaluating the result (Painuli, Elangovan, & Sugumaran, 2014).

3.7. Backpropagation Neural Network Classifier

Neural Networks (NN) are important data mining tool used for classification and clustering. It is an attempt to build machine that will mimic brain activities and be able to learn by examples. If NN is supplied with enough examples, it should be able to perform classification and even discover new trends or patterns in data. The NNs have been efficaciously used for classification purposes in medical domains, including the classification of cancer samples in gene expression data. In this section, a three-layer Backpropagation Neural Network (BPN) is considered as a classifier. The three layers are input, hidden and output layers. Each layer can have number of nodes and nodes from input layer are connected to the nodes from hidden layer. Nodes from hidden layer are connected to the nodes from output layer. Those connections represent weights between nodes. The general architecture of a three-layer BPN with 'm' input neurons, 'n' hidden neurons and one output neuron is given in Figure 1.

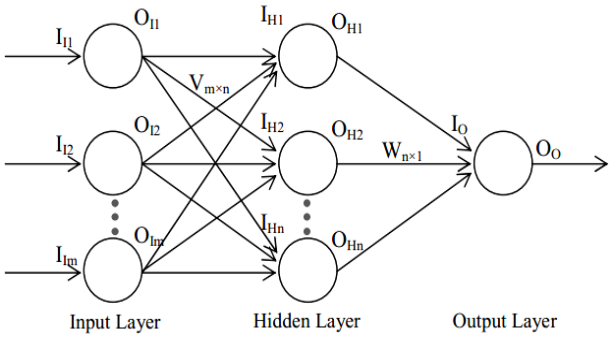

Figure 1. Backpropagation Neural Network

Initially the inputs are normalized between [0, 1]. Two weight matrices are used: the weight matrix V represents the weights of synapses connecting input neurons and hidden neurons and the weight matrix W represents weights of synapses connecting hidden neurons and output neurons. The weights between neurons are initialized between [-0.5, 0.5].

$$[V]0 = [\text{random weights}] \tag{8}$$

$$[W]0 = [\text{random weights}] \quad (9)$$

Initially a training sample is presented to the input layer II as inputs to the input layer. By using linear activation function, the output of the input layer may be evaluated as: OI = II. Then the inputs to the hidden layer are computed by using the output of the input layer and weight V as:

$$IH = VTOI \quad (10)$$

Let the hidden layer units evaluate the output using the purely in transfer function as: OH = IH.

Then in the next step, the inputs to the output layer are obtained by multiplying corresponding weights of synapses as in equation (11)

$$IO = WT\ OH \quad (11)$$

And the output layer units evaluate the output using linear activation function as: OO = IO

From the output of the network, the error is calculated between the network output and the desired output as for the i^{th} training set as: E = |Tk– OOk|, if the error is negligible (≤ 0.0001), then the learning is continued with the next training sample, otherwise, the weights are updated based on the error and the learning is continued with the same training sample until the error becomes low.

The weights between hidden to output layer are back-propagated as:

$$D = (Tk\text{-}OOk)\ OOk(1\text{-}OOk) \quad (12)$$

$$[Y] = OH.d \quad (13)$$

$$[\Delta W] = \alpha[W]t\eta[Y] \quad (14)$$

where α and η are momentum and learning rate parameters to control the weight updation process in BPN architectures. The weights between inputs to hidden layer are backpropagated as:

$$E = [W]\ \{d\} \text{ and } d^* = ei(OH_i)(1\text{-}OH_i) \text{ and} \quad (15)$$

$$[X] = OId^* = IId^* \quad (16)$$

$$[\Delta V] = \alpha[V]t\eta[X] \quad (17)$$

Update the weight matrices as

$$[V]t+1 = [V]t + [\Delta V] \text{ and } [W]t+1 = [W]t + [\Delta W] \quad (18)$$

This learning process is then repeated for all the training samples. As discussed in the classifiers review, the time to train BPN is probably identified as biggest disadvantage. They also require very large sample sets to train model efficiently. Especially for real-world problems which are often characterized by their imbalanced nature (Hala, Own, & Abraham, 2014). The unbalanced distribution of data leads to poor performance of above classification techniques. To overcome these disadvantages and improve the performance of BPN, a novel Weighted Rough Neural Network has been proposed.

In this Paper, a novel method Weighted Rough Neural Network (WRNN) is proposed to handle inconsistent, uncertain and class imbalance dataset.

4. WEIGHTED ROUGH NEURAL NETWORK CLASSIFIER

4.1 Imbalanced data Problem

Imbalanced data means that one of the classes has more samples than the other classes. The class with more samples is called the majority class while the other is the minority class. In many applications the minority class holds the most important information, such as in disease prediction, fraud detection, risk management, natural and physical phenomena, etc. Most classification techniques perform poorly with the minority class. There are three suggested techniques to overcome imbalanced data problems. The first is to create or modify the existing classification algorithms to deal with class imbalance problems. Data re-sampling is the second technique which includes over sampling or under sampling the data set to adjust the size of data set. The last approach is the feature selection, which is used to select a subset of features that allow the classifier to reach optimal performance (Rushi, Snehlata, & Latesh, 2013).

4.2 Weighted Rough Neural Network Algorithm

In WRNN algorithm, the rough set theory is integrated with backpropagation network to classify gene expression data. Rough sets are particularly useful to model an information system IS = (U, A, C, D). The universe U corresponds to the set of instances. The indiscernibility relation R can be modeled by means of the conditional attributes A: instances x and y for which the conditional attributes are highly related will have a high membership degree R(x; y) or will be equivalent for R. The concepts to be approximated correspond to the decision classes that are derived from the decision attribute.

In the traditional rough set, all samples have equal weight without considering the distribution of samples. Here, we apply Class equal Sample Weighting (CSW) scheme (Hala, Own, & Abraham, 2014) to build a weighted decision system WIS = (U, Aw, Cw, D). By using this method, samples belonging to majority class have smaller weight while samples in the minority class have larger weight. Based on the weighted information system we perform the 'inference decision making'. The weighted values are to be discretized since rough set based classification is proposed. The boundary values of weighted information system are considered as uncertain values, and inference decision making is done based on the similarity measure. The similarity measure is evaluated for elements of boundary with the centroid of each class lower approximation, and the decision value is updated according to the closest centroid. Rough set and Neural Networks can solve the complex and high dimensional problems, which are called RNNs (Zhang, 2007) (Rajakeerthana, Velayutham, & Thangavel, 2014) (Zhao & Chen, 2002). A rough neuron can be viewed as a pair of neurons, in which one neuron corresponds to the upper boundary and the other corresponds to the lower boundary. Upper and lower neurons exchange information with each other during the calculation of their outputs. The algorithm of WRNN classifier method is described as follows.

Algorithm 1:	WRNN (C, D)
Input:	IS = (U, A, C, D) be a decision system data,
C - Conditional attributes, D - Decision attribute,	
Output:	Predicted Decision attribute
Step 1:	Set, WIS ← [ ], ND ← [ ]
Step 2:	Do
Step 3:	For every ai ∈ A
Step 4:	For every dj ∈ D
Step 5:	If ai ∈ dj then
Step 6:	$w\_i \leftarrow 1/((n(D) \times n(A\_j)))$
	where, n(D) – No. of decision classes
	n(Aj) – No. of samples classified as dj
Step 7:	Form Weighted Information System,
	WIS ← ai × wi here, WIS = (U,A w,Cw,D)
	end
	end
Step 8:	Discretize WIS using K-Means clustering
Step 9:	Compute the Upper Approximation
	$R\overline{X}w \leftarrow \{x \in U \mid [x]Cw \cap X \neq \Phi\}$

Step 10:	Compute the Lower Approximation $\_RXw \leftarrow \{x \in U \mid [x]Cw \subseteq X\}$	
Step 11:	Compute the Boundary Region $wBNDC(x) \leftarrow \cup R\overline{X}w - \cup \_RXw$	
Step 12:	$wCENd(x) \leftarrow Mean(\_RXw)$	d = 1,2, ..., |D|
Step 13:	For every wBNDC(x)	
Step 14:	For every wCEND (x)	
Step 15:	Dij ← Dist(wCEND(x), wBNDC(x)) here, Dist – Euclidian distance (Equ 5.1) End	
Step 16:	T ←index(min (Dij, wCEND(x))) End	
Step 17:	Update D ← T	
Step 18:	Call BPN(C, D)	

In the above algorithm, wBNDC(x) values are considered as uncertain values, and inference decision making is done based on the similarity measure. The similarity measure is evaluated for every element of wBNDC(x) with the centroid of each class lower approximation wCEND(x), and the decision value is updated according to the closest centroid.

5. PERFORMANCE EVALUATION OF CLASSIFICATION RESULTS

We subjected the classification scheme to a series of tests to assess the performance of the WRNN classification performance. The primary performance assessment involved executing a cross validation with the data.

5.1 Cross Validation

Cross validation is a statistical method that divides data into a fixed n folds, or partitions, of approximately the same size (Witten & Frank, 2005), for example, three fold search containing one-third of the data. The first fold is then used for classifier testing, while the remaining n-1 folds are used for classifier training. The process is repeated so that each of the n folds is used for testing. The resulting process is known as n fold cross-validation. This process can be combined with stratification, or random sampling to create the folds so as to guarantee that each class is represented evenly in both the testing and training sets, which results in approximately the same class distribution across folds. The result is called stratified n fold cross-validation.

Ten folds stratified cross-validation is a standard way of measuring the error rate or accuracy of a learning scheme on a particular dataset (John & Langley, 1995). In this form of cross-validation, the data is divided randomly into ten subsets of approximately the same size. Among the partitions nine of them were used as training set and the remaining one is used as a test set. The classifier is run ten times. Each run, the classifier uses a different subset as the testing set while the other subsets are combined to use as the training set. This results in ten different accuracy / error estimates for the learner that can be averaged to provide an overall accuracy / error estimate. Large test sets gives a good assessment of the classifiers performance and small training sets which result in a poor classifier.

5.2 Classifier Performance Measures

A confusion matrix contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix.

Table : 1 Confusion Matrix For A Two-Class Problem

	Positive prediction	**Negative prediction**
Positive class	True positive (TP)	False negative (FN)
Negative class	False positive (FP)	True negative (TN)

Table 1 shows a confusion matrix for a two-class problem with positive and negative class values. From such a matrix, it is possible to extract a number of widely used metrics to measure the performance of a classifier, such as error rate, defined as in the equation (19).

$$Error = FP+FN/(TP+FN+TN+FP) \quad (19)$$

and overall accuracy is the proportion of the total number of predictions that were correct, it is defined as in the equation (20).

$$Accuracy = TP+TN/(TP+FN+TN+FP) \quad (20)$$

It is possible to derive five performance metrics from Table I to measure the classification performance on the positive and negative classes independently.

- Sensitivity or Recall (True positive rate): It is the proportion of positive cases that were correctly identified, as calculated using the equation (21).

$$Recall = TP/(TP+FN) \quad (21)$$

- Specificity (True negative rate): It is defined as the proportion of negatives cases that were classified correctly, as calculated using the equation (22).

$$Specificity = TN/(TN+FP) \quad (22)$$

- False Positive Rate (FPR): The false positive rate (FPR) is the proportion of negatives cases that were incorrectly classified as positive, as calculated using the equation (23)

$$FPR = FP/(FP+TN) \quad (23)$$

- Precision: Precision is the proportion of the predicted positive cases that were correct, as calculated using the equation (24)

$$Precision = TP/(TP+FP) \quad (24)$$

- F-measure: A measure that combines precision and recall is the harmonic mean of precision and recall. The traditional F-measure or balanced F-score is defined as in the equation (25),

$$Fmeasure = 2*((Precision*Recall)/(Precision+Recall)) \quad (25)$$

The advantages of the above five performance measures are of being independent of class costs and a priori probabilities. The aim of a classifier is to minimize false positive and negative rates, or similarly to maximize true negative and positive rates.

6. EXPERIMENTAL RESULTS

The results provided in Table 3 and Table 4 have been used to investigate the performance of BPN, WEKA classifiers and proposed WRNN. Table 2 indicates selected genes by applying six discretization experiments on Docetaxel treatment Breast cancer gene expression dataset and Table 4 provides selected genes by applying RKMQR on Leukemia gene expression data set. In this Paper, the performance of the eight classifiers is evaluated by using ten fold cross-validations and their measures such as Accuracy, Sensitivity, False Positive Rate, Precision, Recall, F-Measure, Specificity, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

Table 2 shows the selected marker genes from breast cancer gene expression dataset. Breast cancer data set consists of 12,625 genes and 24 samples (14 Resistant, 10 Sensitive). We obtained 9 genes as marker genes out of 12,625 genes by applying discretization experiments. The marker genes are namely 'PSG3', 'RPS17', 'PLD3', ‘MAPK3’, ‘PCBP1’, ‘DUSP1’, ‘SMAD3’, ‘KAT2B’ and ‘RPL3’ which are given as input to the eight classifiers. The results are tabulated in Table 2.

Table : 3 Selected Genes From Breast Cancer Gene Expression Dataset

S.No	Experiments		Selected Genes
1	Entropy	'PSG3'	'PSG3', 'RPS17', 'PLD3', ‘MAPK3’, ‘PCBP1’, ‘DUSP1’, ‘SMAD3’, ‘KAT2B’, ‘RPL3’
2	Rough	'RPS17', 'PLD3'	
3	Naïve	‘MAPK3’	
4	Max-Min	‘MAPK3’, ‘PCBP1’	
5	K-Means	‘DUSP1’, ‘SMAD3’	
6	EWI	‘KAT2B’, ‘RPL3’	

Table : 4 Experimental Result of Each Classifier Using Breast Cancer Dataset

Classifiers	WRNN	BPN	Naïve Bayes	JRip	J48	RF	Decision Table	K-Star
Accuracy	95.83	91.67	83.33	91.67	91.67	70.83	87.50	58.33
Sensitivity	95.80	91.70	83.30	91.70	91.70	70.80	87.50	58.30
FPrate	05.80	08.80	11.90	08.80	06.00	40.80	08.90	58.30
Precision	96.10	91.70	88.10	91.70	93.10	80.60	90.40	34.00
Recall	95.80	91.70	83.30	91.70	91.70	70.80	87.50	58.30
F-Measure	95.80	91.70	83.30	91.70	91.70	65.90	87.60	43.00
Specificity	94.20	91.20	88.10	91.20	94.00	59.20	91.10	41.70
MAE	04.17	11.67	16.67	06.67	11.87	29.17	12.50	48.91
RMSE	20.41	23.27	40.82	23.80	27.91	54.01	35.36	49.47

The experimental results show that the highest correctly classified instances is 23 (95.83%) out of 24 samples by WRNN and the lowest correctly classified instances is 14 (58.33%) by K-Star algorithm. The BPN, Naïve Bayes, JRip, J48, Random Forest and Decision Table classifiers produces 91.67%, 83.33%, 91.67%, 91.67%, 70.83% and 87.50% accuracy respectively. In fact, the highest accuracy (95.83%), which is belongs to proposed WRNN classifier, followed by BPN, JRip and J48 classifier. They produce same accuracy (91.67%) rate. Minimum false positive rate 5.8% is achieved from WRNN classifier and highest false positive rate 58.30% recognized from K-Star. The highest true positive rate or sensitivity (95.80%) and Specificity (94.20%) are obtained from WRNN.

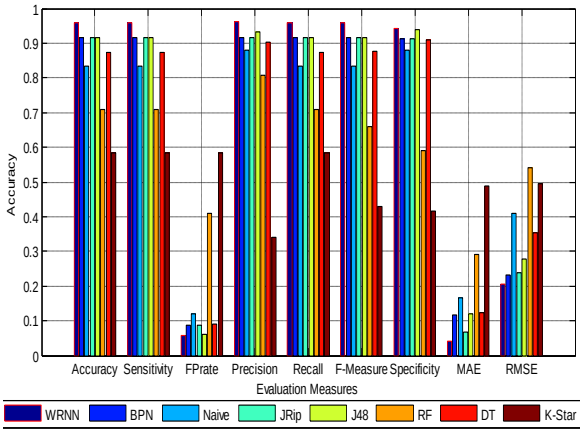

Figure 2. Performance measures of each classifier using breast cancer dataset

From Table 3 and Figure 2, the confusion matrix of each classifier can be observed which shows nine evaluation measures. One can observe that the differences of errors resultant from the training of the eight selected classifier algorithms. This experiment implies a very commonly used indicator which is MAEs and RMSEs. The average of MAE and RMSE of all algorithms are ranging from 4.17 to 49.47. The proposed WRNN gives the lowest mean absolute error 4.17 and lowest root mean squared error 20.41 when we compared to other methods. An algorithm which has a lower error rate and maximum accuracy will be preferred as it has more powerful classification capability. Taking mean absolute error, false positive rate and classification accuracy WRNN is considered as the best classification algorithm.

Table 4 shows that the selected marker genes from leukemia gene expression dataset. Leukemia gene expression data set consists of 7129 genes and 38 bone marrow samples (27 ALL, 11 AML). We obtained 8 genes out of 7129 genes as marker genes by applying RKMQR. The marker genes are #56, #3252, #930, #1674, #1884, #865, #1902 and #587 which are inputs of eight classifiers in following experiment.

Table : 5 Selected Genes From Leukemia Gene Expression Dataset

Method	Experiments		Selected Genes
RKMQR	*K*=5	#56, #3252	#56, #3252, #930, #1674, #1884, #865, #1902, #587
	K=7	#930, #3252, #1674, #1884	
	K=10	#865, #1902, #587, #3252	

Table : 6 Experimental Result of Each Classifier Using Leukemia Dataset

Classifiers	WRNN	BPN	Naïve Bayes	JRip	J48	RF	Decision Table	K-Star
Accuracy	97.05	97.05	91.17	94.11	91.17	85.29	88.23	82.35
Sensitivity	92.85	100.00	85.71	92.85	92.85	78.57	92.85	78.57
FPrate	00.00	05.00	05.00	05.00	10.00	10.00	15.00	15.00
Precision	100.00	93.33	92.30	92.85	86.67	84.62	81.25	78.57
Recall	92.85	100.00	85.71	92.85	92.85	78.57	92.85	78.57
F-Measure	96.29	96.55	88.89	92.85	89.65	81.48	86.67	78.57
Specificity	100.00	95.00	95.00	95.00	90.00	90.00	85.00	85.00
MAE	01.25	11.87	12.50	05.67	11.67	24.17	29.17	36.91
RMSE	11..87	28.91	35.36	26.10	23.27	44.06	54.01	39.46

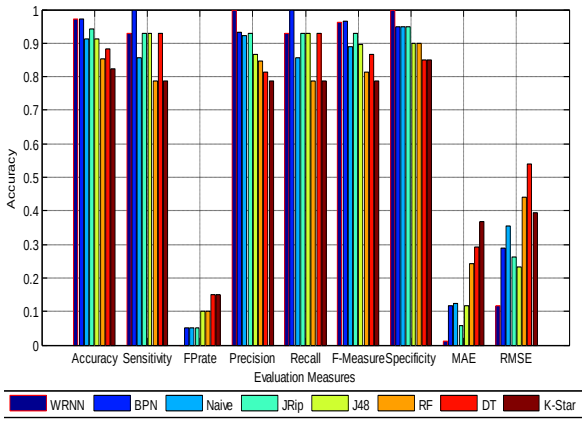

Figure 3. Performance measures of each classifier using Leukemia dataset

From Table 5 the confusion matrix of each classifier can be observed which shows nine evaluation measures. The results are obtained using leukemia gene expression data set. The experimental results show that the highest correctly classified instances is 33 (97.06%) out of 34 samples by WRNN and the lowest correctly classified instances is 28 (82.35%) by K-Star algorithm. The BPN, Naïve Bayes, JRip, J48, Random Forest and

Decision Table methods produced 97.05%, 91.17%, 94.11%, 91.17%, 85.29% and 88.23% accuracies respectively. WRNN and BPN have produced the highest classification accuracy than the other classification methods. Though both the same accuracy WRNN the lowest mean absolute error compared to BPN. If both the algorithm has the same error rate and accuracy then root mean squared error and false positive rate is taken into consideration.

The aim of a high-quality classifier model is to minimize the false positive rate. Compare with other methods, Minimum false positive rate 0.0% is achieved from WRNN classifier and highest false positive rate 15.00% recognized from Decision Table and K-Star. The highest true positive rate or sensitivity 100% is achieved by BPN and Specificity 100% is achieved by WRNN. From Fig 3, it is observed that the differences of errors resultant from the training of the eight selected algorithms. The average of MAE and RMSE of all algorithms are ranging from 1.25 to 54.01. To compare with other methods, proposed WRNN produces the lowest mean absolute error 1.25 and lowest root mean squared error 11.87. The proposed WRNN produces the best accuracy when we considered absolute error, false positive rate and classification accuracy. From the above two experimental results we obtained that the WRNN clearly outperform than the conventional neural networks and other classifier methods.

7. CONCLUSION

This Paper has reviewed and studied several classification methods and proposed a novel WRC classification method to handle inconsistent and class imbalance dataset. The effectiveness of WRC model has been compared with a conventional BPN and WEKA classifiers. The performances of the classifiers are evaluated by using ten fold cross-validations with the suitable measures. The estimated errors obtained from WRNN model are significantly lower than the other models. Experimental analysis shows that the novel WRNN algorithm is effective and suitable for evaluating the classification. The proposed WRC produced the highest classification accuracy and Minimum false positive rate. Further, experimental studies showed that WRC based technique outperformed, compared with the existing techniques.

ACKNOWLEDGMENT

The present work is supported by Special Assistance Programme of University Grants Commission, New Delhi, India (Grant No. F.3-50/2011(SAP-II).

REFERENCES

Breiman, L. (2001). Random forests. Machine Learning, 45 (1), 5 – 32.

Cleary, J., & Trigg, L. (1995). K*: An Instance-based Learner Using an Entropic Distance Measure. In Proceedings of the 12th International Conference on Machine Learning, 108–114.

Devasena, C. L., Sumathi, T., Gomathi, V. V., & Hemalatha, M. (2011). Effectiveness Evaluation of Rule Based Classifiers for the Classification of Iris Data Set. Bonfring International Journal of Man Machine Interface, 5 – 9.

Fauset, L. (1994). Fundamentals of Neural Network Architecture, Algorithms and Applications. Prentice Hall.

Forbes, A. (1995). Classification Algorithm evaluation: performance measure based on confusion matrix. 11, 189-206.

Hala, S., Own, & Abraham, A. (2014). A Novel-weighted Rough Set-based Meta Learning for Ozone Day Prediction. Acta Polytechnica Hungarica, 11 (4), 59 – 78.

Han, J., & Kamber, M. (2001). Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers.

John, G. H., & Langley, P. (1995). Estimating Continuous Distributions in Bayesian Classifiers. Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, 139 – 157.

Kavitha, C. R., & Mahalekshmi, T. (2015). A Comparative Study of Classification Algorithms on Aliphatic Carboxylic Acids Data Set using WEKA. International Journal of Emerging Science and Engineering (IJESE), 3 (7), 319–328.

Painuli, S., Elangovan, M., & Sugumaran, V. (2014). Tool condition monitoring using K-star algorithm. Journal Expert Systems with Applications: An International Journal archive, 41 (6), 2638-2643.

Rajakeerthana, K. T., Velayutham, C., & Thangavel, K. (2014). Mammogram Image Classification Using Rough Neural Network. ICC3. Advances in Intelligent Systems and Computing Series, Springer, 133-138.

Rushi, L., Snehlata, S. D., & Latesh, M. (2013). Class Imbalance Problem in Data Mining: Review. International Journal of Computer Science and Network (IJCSN), 2, 226-230.

Salzberg, S. L. (1994). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, Inc.

Sharma, T. C., & Jain, M. (2013). WEKA Approach for Comparative Study of Classification Algorithm. International Journal of Advanced Research in Computer and Communication Engineering, 2 (4), 95 - 101.

Thangaraj, M., & Vijayalakshmi, C. R. (2013). Performance Study on Rule based Classification Techniques across Multiple Database Relations. International Journal of Applied Information Systems (IJAIS), 5 (4), 1 – 7.

Tiwary, D. K. (2014). A Comparative study of classification algorithms for credit card approval using WEKA. International Interdisciplinary Research Journal, 2 (3), 165 – 174.

Witten, I. H., & Frank, E. (2005). Data Mining: Practical Machine Learning Tools and Techniques (2 ed.). Morgan Kaufman.

Zhang, D. (2007). Integrated methods of rough sets and neural network and their applications in pattern recognition. 256 – 272.

Zhao, W., & Chen, G. (2002). A survey for the integration of rough set theory with neural networks. Systems engineering and electronics, 24 (10), 103-107.