SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
VOLUME 3, ISSUE 2
ISSN 2054-7390

A Novel Approach to Compute Confusion Matrix for Classification of n-Class Attributes with Feature Selection

V. Mohan Patro[1] and Manas Ranjan Patra[2]
Department of Computer Science, Berhampur University, Berhampur, Odisha, India
[1]vmpatro@gmail.com, [2]mrpatra12@gmail.com

ABSTRACT

Confusion matrix is a useful tool to measure the performance of classifiers in their ability to classify multi-classed objects. Computation of classification accuracy for 2-classed attributes using confusion matrix is rather straightforward whereas it is quite cumbersome in case of multi-class attributes. In this work, we propose a novel approach to transform an n × n confusion matrix for n-class attributes to its equivalent 2 × 2 weighted average confusion matrix (WACM). The suitability of WACM has been shown for a classification problem using a web service data set. We have computed the accuracy of four classifiers, namely, Naïve Bayes (NB), Genetic Programming (GP), Instance Based Lazy Learner (IB1), and Decision Tree(J48) with and without feature selection. Next, WACM has been employed on the confusion matrix obtained after feature selection which further improves the classification accuracy.

Key words: Confusion Matrix, Classifiers, Feature Selection, Weighted Average Confusion Matrix, Classification Accuracy, Weighted average accuracy.

1 Introduction

Confusion matrix provides the basis for evaluating the performance of any classifier with the help of its four components, viz., True Positive (TP), False Negative (FN), False Positive (FP) and True Negative (TN). Among others, classification accuracy is the major parameter to judge the efficiency of a classifier. Classification accuracy of a classifier on a given data set refers to the percentage of test set tuples that are correctly classified by the classifier. It reflects how well the classifier recognizes tuples of various classes. The error rate or misclassification rate of a classifier M can be expressed as 1- Acc(M), where Acc(M) is the accuracy of M [1].

The common form of expressing classification accuracy is the error matrix (confusion matrix or contingency table). Error matrices compare the relationship between the known reference data and the corresponding results of classification on a class-by-class basis. The overall accuracy is computed by dividing the total number of correctly classified elements (the sum of the elements along the major diagonal) by the total number of elements in the confusion matrix. However, there are other contributing elements in the true negative component (which is an n-1 × n-1 matrix) of the confusion matrix which are ignored while computing the overall accuracy. This considerably reduces the accuracy of a classifier. The proposed WACM considers the contribution of those left out elements which eventually increases the accuracy of a classifier.

DOI: 10.14738/tmlai.32.1108
Publication Date: 28th April, 2015
URL: http://dx.doi.org/10.14738/tmlai.32.1108

In our earlier work [2], we have applied the weighted average technique for computing classification accuracy wherein the individual classification accuracy for each class of a multi-classed attribute is calculated first and then the individual accuracies of the respective classes are aggregated using the weighted average accuracy algorithm. This method has a limitation as it only helps in computing the classification accuracy. But, in order to calculate other performance criteria like sensitivity or true positive rate (TPR), specificity (SPC) or True Negative Rate, precision or positive predictive value (PPV), negative predictive value (NPV), fall-out or false positive rate (FPR), false discovery rate (FDR), Miss Rate or False Negative Rate (FNR), accuracy (ACC), F1 score, Matthews correlation coefficient (MCC), Informedness, Markedness etc. the four components, namely, TP, FN, FP and TN of a confusion matrix plays a vital role. In this work, we have proposed a technique to build a weighted average confusion matrix and have shown its novelty in the performance evaluation of four different classifiers, namely, Naïve Bayes (NB), Genetic Programming (GP), Instance Based Lazy Learner (IB1), and Decision Tree(J48). It is shown that the proposed approach considerably enhances the accuracy.

2 Weighted Average Confusion Matrix

2.1 Confusion Matrix

A confusion matrix (also known as a contingency table or an error matrix) is a table layout that allows visualization of the performance of a supervised learning algorithm [3]. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. All correct guesses are located along the diagonal of the table such that errors can be easily visualized by any non-zero values outside the diagonal. In order that a classifier yields better accuracy it is necessary the total number of instances of a particular class in a data set should be represented along the diagonal of the confusion matrix (CM) as far as possible.

2.2 Confusion Matrix for two classes

Here we consider an example of a Two-class scenario that depicts the buying behavior of customers. Let us consider an attribute “buys computer” which can take two possible values “yes” and “no”. Next, we introduce the notion of positive tuples when the class attribute value is “yes” (i.e., buys computer = “yes”) and negative tuples when the class attribute value is “no” (e.g., buys computer = “no”). True positives refer to the positive tuples that were correctly labeled by the classifier as positive, while true negatives are the negative tuples that were actually labeled as negative by the classifier. False positives are the negative tuples that were incorrectly labeled (e.g., tuples of class buys computer = “no” for which the classifier predicted buys computer = “yes”). Similarly, false negatives are the positive tuples that were incorrectly labeled (e.g., tuples of class buys computer = “yes” for which the classifier predicted buys computer = “no”).

Table 1: Confusion matrix for 2-class scenario

		Predicted Class	
		C_1	C_2
Actual Class	C_1	True positive	False negative
	C_2	False positive	True negative

C_1 – particular class C_2 – different class

True positive (TP) - The number of instances correctly classified as C1

False negative (FN) - The number of instances incorrectly classified as C2 (actually C1)

False positive (FP) - The number of instances incorrectly classified as C1 (actually C2)

True negative (TN) - The number of instances correctly classified as C2

P = Actual positive = TP + FN

P1 = Predicted positive =TP + FP

N = Actual negative = FP + TN

N1 = Predicted negative =FN + TN

TP rate = Sensitivity = TP / P = Recall

TN rate = Specificity = TN / N

FP rate = selectivity =1 – TN rate = FP / N

Precision = TP / P1

Accuracy = (TP + TN) / (P + N)

= TP / (P + N) + TN / (P + N)

= TP / P × P / (P + N) + TN / N * N / (P + N)

= Sensitivity × P / (P + N) + Specificity × N / (P + N)

$$\text{F1 score} = \frac{2 \times precision \times recall}{precision + recall}$$

2.3 Conversion into 2 × 2 confusion matrix

If a classification system has been trained to distinguish between cats, dogs and rabbits, a confusion matrix will summarize the results of testing the algorithm for further inspection [3]. Assuming a sample of 27 animals — 8 cats, 6 dogs, and 13 rabbits, the resulting confusion matrix could look like the table 2.

Table 2: Confusion matrix for a three class scenario

		Predicted class		
		Cat	Dog	Rabbit
Actual class	Cat	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

In this confusion matrix, of the 8 actual cats, the system predicted that three were dogs, and of the six dogs, it predicted that one was rabbit and two were cats. Considering the confusion matrix above, the corresponding table of confusion (Ref. Table 3), for the cat class, would be:

Table 3: Table of confusion for the class "Cat"

5 true positives (actual cats that were correctly classified as cats)	3 false negatives (cats that were incorrectly marked as dogs)
2 false positives (dogs that were incorrectly labelled as cats)	17 true negatives (all the remaining animals, correctly classified as non-cats)

Likewise, one can obtain 2 × 2 matrices for dog and rabbit classes. The actual count of a particular class is taken as the weight for the same class, e.g., for the "Cat" class the actual count is 8, hence the weight for the "Cat" class is taken as 8 at the time of building the WACM. Aggregating all the individual confusion matrices along with the weights of the individual classes, the weighted average confusion matrix for an attribute is calculated. The process of aggregation is presented in the following algorithm.

Weighted Average Confusion Matrix Algorithm:
Input: n × n Confusion Matrix
Output: Matrix containing elements of Weighted Average Confusion Matrix
WACM (A, CM)
//CM is n × n confusion matrix, where n is number of classes of an attribute, on which basis we //calculate classification accuracy. A is n+1 × n+5 matrix, where first n × n is filled up with CM
Begin
For i=1 to n, A (n+1,i) = $\sum_{j=1}^{n} A(j,i)$ // sum of n columns
For i=1 to n+1, A(i,n+1) = $\sum_{j=1}^{n} A(i,j)$ // sum of n+1 rows, where A(n+1,n+1) is number of instances
For i=1 to n, A(i,n+2) = A(i,i) // A(i,n+2) is TP of individual class
For i=1 to n, A(i,n+3) = A(i,n+1) - A(i,n+2) // A(i,n+3) is FN of individual class
For i=1 to n, A(i,n+4) = A(n+1,i) - A(i,n+2) // A(i,n+4) is FP of individual class
For i=1 to n, A(i,n+5) = A(n+1,n+1) - $\sum_{j=n+2}^{n+4} A(i,j)$ //A(i,n+5) is TN of individual class
For i=n+2 to n+5, A(n+1,i) = $\sum_{j=1}^{4} [A(j,n+1) * A(j,i)]$ / A(n+1,n+1)
// A(n+1,n+2)….A(n+1,n+5) are 4 components(TP, FN, FP, TN) of 2 × 2 target confusion matrix
//Here A(j,n+1) is weight & $\sum_{j=n+2}^{n+5} A(n+1,j)$ is seen that it equals to A(n+1,n+1)
Return **A**
End

In the subsequent sections we show the applicability of WACM in improving the classification accuracy of classifiers. For our experimentation we have considered four different classifiers, namely, Naïve Bayes (NB), Genetic Programming (GP), Instance Based Lazy Learner (IB1), and Decision Tree(J48). First, we determine the classification accuracy of the individual classifiers and then apply feature selection to observe the improvement in accuracy. Finally, WACM is applied to compare the accuracy so obtained with the earlier experiments.

3 Classification Techniques

3.1 Naïve Bayesian Classifier

The Naïve Bayesian Classifier is one of the Bayesian Classifiers [4] which has been extensively used for classifying objects with a higher degree of accuracy. It has proven performance in various Machine Learning and Data Mining applications [5] - [8]. Naïve Bayesian classifiers assume that, given the class label all attributes within the same class are independent. Based on this assumption, the Naïve Bayesian classification rule is expressed as:

$$P(C|E) = \arg\max_c P(C) \prod_{i=1}^{n} P(A_i|\, C)$$

Where C represents a class label, Ai the attributes, and E the unclassified test instance. E is classified into class C with the maximum posterior probability.

3.2 Genetic Programming (GP)

It is a specialization of genetic algorithm (GA) [9]. Genetic Algorithm (GA) is a global search method based on natural selection procedure consisting of genetic operators such as selection, crossover and

mutation. GA optimizers are particularly effective in a high-dimension, multi-modal function, in which the number of variables tend to be higher. GA performs its searching process via population-to-population (instead of point-to-point) search. A member in a population called a chromosome is represented by a binary string comprising of 0 and 1 bits. Bits of the chromosome are randomly selected and the length of bit strings is defined in relevance. However, real values are taken in continuous genetic algorithm. In order to apply the methodology, a randomly generated initial population is required. From initial population, child population is born guided by three operators such as reproduction, crossover and mutation. New born child members are judged by their fitness function values. These child members act as parents in the next iteration. This procedure is repeated till the termination criteria are met.

The pseudo code of a genetic algorithm is depicted as below.

```
Simple Genetic Algorithm ( )
{
        Initialize the Population;
        Calculate Fitness function;
        While (Fitness Value! = Optimal Value)
        {
                Selection;
                Crossover;
                Mutation;
                Calculate fitness Function;
        }
}
```

3.3 Lazy Learner (IB1)

IB1 is a nearest-neighbor classifier [10] which uses normalized Euclidean distance to find the training instance closest to the given test instance, and predicts the same class as this training instance. If multiple instances have the same (smallest) distance to the test instance, the first one found is used.

The Euclidean distance between two points or tuples, say X1 = (x11, x12, …..x1n) and X2 = (x21, x22, …. x2n) is

$$\text{Dist}(X_1,X_2) = \sqrt{\sum_{i=1}^{n}(x_{1i} - x_{2i})^2}$$

IB1 or IBL (Instance-Based Learning) is a comprehensive form of the Nearest Neighbor algorithm [10, 11]. IB1 generates classification predictions using only specific instances. Unlike nearest neighbor algorithm, IB1 normalizes the range of its attributes, processes instances incrementally and has a simple policy for tolerating missing values [10]. IB1 uses simple normalized Euclidean distance (similarity) function to yield graded matches between training instance and given test instance [11].

3.4 Decision Tree (J48)

J48 is a classifier for generating a pruned or unpruned C4.5 decision tree [12]. J48 is an open source Java implementation of the C4.5 algorithm [13] in the WEKA data mining tool. C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. The training

data is a set S = s1, s2, ….. of already classified samples. Each sample si consists of a p-dimensional vector (x1,i, x2,i, …, xp,i) where the xj represent attributes or features of the sample, as well as the class in which si falls.

4 Experimental Set Up

4.1 Data Set

The QWS (Quality of Web Service) dataset [14-16] consists of data from over 5000 web services out of which the public dataset consists of a random 364 web services. The service descriptions were collected using the Web Service Crawler Engine (WSCE) [17]. The majority of Web services were obtained from public sources on the Web including Universal Description, Discovery, and Integration (UDDI) registries, search engines, and service portals. The public dataset consists of 364 web services each with a set of nine Quality of Web Service (QWS) attributes that have been measured using commercial benchmark tools. Each service was tested over a ten-minute period for three consecutive days. WSRF is used to measure the quality ranking of a web service based on the nine quality parameters (1-9 in Table-4)

Table 4: QWS Parameter description

P-ID	Parameter Name	Description	Units
1	Response Time (RT)	Time taken to send a request and receive a response	ms
2	Availability (AV)	Number of successful invocations/total invocations	%
3	Throughput (TP)	Total Number of invocations for a given period of time	Invokes per second
4	Success ability (SA)	Number of responses / number of request messages	%
5	Reliability (REL)	Ratio of the number of error messages to total messages	%
6	Compliance (CP)	The extent to which a WSDL document follows WSDL specification	%
7	Best Practices (BP)	The extent to which a Web service follows WS-I Basic Profile	%
8	Latency (LT)	Time taken for the server to process a given request	ms
9	Documentation (DOC)	Measure of documentation (i.e. description tags) in WSDL	%
10	WSRF	Web Service Relevancy Function: a rank for Web Service Quality	%
11	Service Classification	Levels representing service offering qualities (1 through 4)	Classifier
12	Service Name	Name of the Web service	None
13	WSDL Address	Location of the Web Service Definition Language (WSDL) file on the Web	None

In table 4, the service parameters 1-9 are used for computation of classification accuracy with respect to four “Service Classification” values, namely, “Platinum” (high quality), “Gold”, “Silver” and “Bronze” (low quality) equivalent to 1 through 4 respectively. Thus, a classifier can give rise to a 4×4 confusion matrix.

4.2 WEKA Workbench

We have used the WEKA (Waikato Environment for Knowledge Analysis) machine learning platform [18] for our experimentation. The WEKA workbench consists of a collection of implemented popular learning schemes that can be used for practical data mining and machine learning.

4.3 Cross-Validation

We employ the cross-validation technique to calculate the accuracy of the classifiers. Cross-validation calculates the accuracy of the model by separating the data into two different subsets, namely, training

set and validation set or testing set. The training set is used to perform the analysis and the validation set is used to validate the analysis. This testing process is continued k times to complete the k-fold cross validation procedure. We have used 10-fold cross-validation wherein the dataset is partitioned into 10 subsets, of which 9 subsets are used as the training fold and a single subset is used as the testing data. The process is repeated 10 times such that each subset is used as a test subset once. The estimated accuracy is the mean of the estimates for each of the classifiers.

4.4 Feature Selection

An attribute selection measure is a heuristic for selecting relevant attributes and reducing redundant and irrelevant attributes in the dataset to improve upon classification accuracy. Therefore, suitable attribute selection method for selecting the most prominent features from the dataset is of paramount importance to enhance the performance of classification accuracy and reduce the computation time. In this study, we have applied two feature selection techniques, namely, Information Gain Attribute Evaluator and Gain Ratio Attribute Evaluator.

4.4.1 Information Gain

It evaluates the worth of an attribute by measuring the information gain with respect to a class. Information gain measure is used to determine how accurately a particular attribute classifies the training data. Information gain is based on the concept of entropy which is widely used in the Information theory domain.

Let node N represents the tuples of partition D. The attribute with the highest information gain is chosen as the splitting attribute for node N. This attribute minimizes the information needed to classify tuples in the resulting partitions and reflects the least randomness or impurity in these partitions [1].

The expected information needed to classify a tuple in D is given by

$$\text{Info (D)} = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

where pi is the probability that an arbitrary tuple in D belongs to class Ci and is estimated by |Ci,D| / |D|. Info(D) is the average amount of information needed to identify the class label of a tuple in D.

$$\text{Info}_A\text{ (D)} = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \text{Info (D}_j\text{)}$$

The term (|D_j |)/(|D|) acts as the weight of the j-th partition. InfoA (D) is the expected information required to classify a tuple from D based on the partitioning by A. Information gain is defined as the difference between the original information requirement and new information requirement. That is

$$\text{Gain (A)} = \text{Info(D)} - \text{InfoA(D)}$$

Using Information Gain Evaluation with Ranker Search method for web service data, top 3 attributes (WSRF, WSDL Address, Service Name) are selected for classification.

4.4.2 Gain Ratio

It evaluates the worth of an attribute by measuring the gain ratio with respect to the class.It applies a kind of normalization to information gain using a “split information” value. The split information value

represents the potential information generated by splitting the training data set D into v partitions corresponding to v outcomes on attribute A, and is expressed as [1]:

$$\text{SplitInfo}_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

The gain ratio is defined as

$$\text{GainRatio(A)} = \frac{\text{Gain(A)}}{\text{SplitInfo(A)}}$$

The attribute with the maximum gain ratio is selected as the splitting attribute.

Using Gain Ratio Evaluation with Ranker Search method for web service data, top 3 attributes (WSRF, Throughput, Response Time) are selected for classification.

5 Implementation of WACM Algorithm

For the chosen attribute "Service Classification" in the Web services data set the confusion matrix is a 4×4 matrix as the attribute can assume 4 possible class values, namely, 'Platinum', 'Gold', 'Silver' and 'Bronze'. Table-5, 6, 7 and 8 depict the 4×4 confusion matrix obtained using Naïve Bayes classifier for the attribute "Service Classification".

The intersection of 1st row and 1st column gives the TP value for the 'Platinum' class. Sum of the remaining elements in the 1st row gives the FN value and sum of the remaining elements in the 1st column gives the FP value for the 'Platinum' class. Similarly, sum of the remaining elements in the entire matrix gives the TN value for the 'Platinum' class. This is shown in table-5 and table-9(as described in table-3). Similarly, for 'Gold', 'Silver' and 'Bronze' class values the "2ndrow and 2nd column", "3rd row and 3rd column", "4th row and 4th column" are respectively consideration for determining the TP, FN, FP, and TN values. Instances are shown in table-6, table-7 and table-8 respectively.

Table 5: Instance for "Platinum"

41	0	0	0
1	94	5	0
0	1	119	0
0	0	0	103

Table 6: Instance for "Gold"

41	0	0	0
1	**94**	5	0
0	1	119	0
0	0	0	103

Table 7: Instance for "Silver"

41	0	0	0
1	94	5	0
0	1	**119**	0
0	0	0	103

Table 8: Instance for "Bronze"

41	0	0	0
1	94	5	0
0	1	119	0
0	0	0	**103**

Table-9 Table of confusion for the class "Platinum"

41 true positives (actual platinum that were correctly classified as platinum)	0 false negatives
1 false positive (gold that were incorrectly classified as platinum)	322 true negatives (all the remaining 'service classification' classes, correctly classified as non-platinum)

Next, we explain the process of obtaining the elements of Table 11.

The available 4 × 4 matrix is the actual data from the input 4 × 4 confusion matrix. First 4 elements of the 5th row represent the sum of column elements and the 5th column the sum of row elements. First 4 elements of the 6th column are for true positive values i.e. individual diagonal elements. First 4 elements of 7th column are for false negative values, which are obtained by subtracting TP from the concerned row sum. First 4 elements of 8th column are for false positive values, which can be obtained by subtracting TP from the concerned column sum. Lastly, first 4 elements of 9th column are for true negative values, which can be obtained by subtracting sum of the TP, FN, and FP from the total number of elements in the confusion matrix. In this way first 4 rows of columns 6, 7, 8 and 9 give TP, FN, FP and TN values for 'Platinum', 'Gold', 'Silver' and 'Bronze' classes respectively as shown in Table-10.

Table 10: Four 2 × 2 confusion matrices

Platinum		**Gold**	
41	**0**	**94**	**6**
1	**322**	**1**	**263**
Silver		**Bronze**	
119	**1**	**103**	**0**
5	**239**	**0**	**261**

Now to aggregate these individual confusion matrices to generate the resultant 2 × 2 confusion matrix, we have taken the actual number of instances for each class as the weight. As per the WACM algorithm, four components (TP, FN, FP and TN) of the WACM are calculated and placed in the last four cells of the last row of Table-11.

Table 11: (n+1) × (n+5) matrix as in algorithm

	Input confusion matrix				Row Sum	TP	FN	FP	TN
Platinum	41	0	0	0	41	41	0	1	322
Gold	1	94	5	0	100	94	6	1	263
Silver	0	1	119	0	120	119	1	5	239
Bronze	0	0	0	103	103	103	0	0	261
Column Sum	42	95	124	103	364	98.81868132	1.978021978	2.035714286	261.1675824

6 Results and Discussion

As explained in section 4, the four classifiers are first applied to classify the web services data set. Next, the same classifiers are used after applying two feature selection methods and the results are analyzed for possible improvement in the degree of accuracy. Finally, the process is repeated by applying the proposed WACM algorithm.

The sum of the 4 aggregated components TP, FN, FP and TN (the last row of table-11) turns out to be 364, which is the total number of instances in the data set. Classification accuracy i.e. (TP+TN) / (TP+FN+FP+TN) for the multi-classed attribute "Service Classification" is same as that calculated using the weighted average accuracy algorithm proposed in our earlier work [2] which establishes correctness of WACM.

Table 12: Classification accuracy for classifiers using IG

Classifier	Without feature selection	With feature selection (IG)	IG + WACM
NB	0.81044	0.980769	0.988973252
GP	0.983516	0.989011	0.993954535
IB1	0.717033	0.947802	0.969659461
J48	**0.997253**	**0.997253**	**0.998935817**

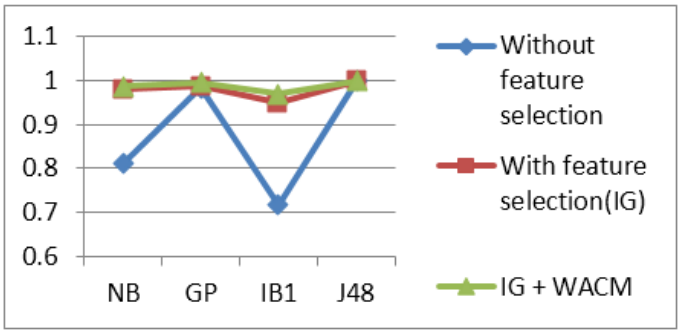

Figure 1: Improvement trend of classifiers

The values in Table-12 and 13 clearly indicate that by applying WACM the performance of the individual classifiers improves to a considerable extent irrespective of the feature selection method used. However, it is observed that both the feature selection methods do not have any impact on the J48 classifier, i.e., the classification accuracy remains unaltered with and without feature selection. Further, the classifier J48 outperforms all other classifiers in terms of accuracy. The improvement trends in both the cases are shown in figure-1 and 2.

Table 13: Classification accuracy of classifiers using GR

Classifier	Without feature selection	With feature selection (GR)	GR + WACM
NB	0.81044	0.887363	0.935824478
GP	0.983516	0.994505	0.997871634
IB1	0.717033	0.93956	0.96441402
J48	**0.997253**	**0.997253**	**0.998935817**

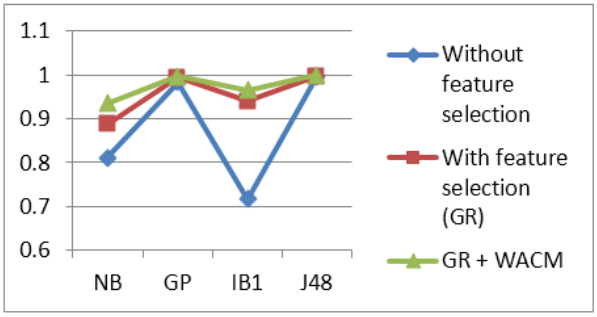

Figure 2: Improvement trend of classifiers

Further, the values for precision, recall and f-value are also computed using the weighted accuracy confusion matrix and are tabulated in table-14. Here also it is observed that the classifier J48 performs much better compared to the rest of the classifiers.

Table 14: Precision, Recall, F-value for the classifiers using WACM

Feature Selection	Classifier	Precison	Recall	F-Value
Information Gain	NB	0.979815314	0.980376124	0.980095639
	GP	0.990139576	0.988007632	0.989072455
	IB1	0.945970296	0.944371763	0.945170354
	J48	**0.998880729**	**0.997274462**	**0.998076949**
Gain Ration	NB	0.891105869	0.875197602	0.883080096
	GP	0.997757848	0.994548923	0.996150801
	IB1	0.935709808	0.935786318	0.935748062
	J48	**0.998880729**	**0.997274462**	**0.998076949**

7 Conclusion

In this work, we have introduced the notion of weighted average confusion matrix which is an aggregation of n number of 2 x 2 confusion matrices each referring to a particular class. Such a matrix enables one to compute the TP, FN, FP and TN components succinctly based on which the performance measures like Precision, Recall, F-value, etc. can be computed more accurately. In order to verify the usability of WACM, we have applied it for calculating the classification accuracy of four classifiers, namely, Naïve Bayes, Genetic Programming, Instance Based Lazy Learner, and Decision Tree. Feature selection techniques are also used to improve the accuracy. A systematic study shows that the performance of each of the classifier is improved to a considerable extent by using the weighted average confusion matrix. In future, we propose to study the impact of increasing the number of data instances on the accuracy level.

ACKNOWLEDGEMENTS

We would like to thank Dr. E. Al-Masri and Dr. Q.H. Mahmoud for providing us with the QWS dataset, which we have used for our experimentation.

REFERENCES

[1]. Han, J., and Kamber, M., 2006, Book on" Data Mining: Concepts and Techniques", 2nd ed., Morgan Kaufmann Publishers, March 2006, ISBN 978-1-55860-901-3.

[2]. Patro, V. M., and Patra, M. R., 2014, "Augmenting Weighted Average with Confusion Matrix to Enhance Classification Accuracy", Transactions on Machine Learning and Artificial Intelligence, Volume 2 No 4, Aug (2014); pp: 77-91

[3]. http://en.wikipedia.org/wiki/Confusion_matrix last accessed on 10/11/14

[4]. Jensen, F.V., 1993, "Introduction to Bayesian Networks". Denmark: Hugin Expert A/S, 1993.

[5]. Wang, Z., and Webb, G. I., 2002, "Comparison of lazy bayesian rule and tree-augmented bayesian learning", IEEE, 2002, pp. 490 – 497.

[6]. Shi, Z., Huang, Y., and Zhang, S., 2005, "Fisher score based naive Bayesian classifier", IEEE, 2005, pp. 1616-1621.

[7]. Xie, Z., and Zhang, Q., 2004, "A study of selective neighborhood-based naïve bayes for efficient lazy learning", 16th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2004.

[8]. Santafe, G., Loranzo, J.A., and Larranaga, P., 2006, "Bayesian model averaging of naive bayes for clustering", IEEE, 2006, Page(s) 1149 – 1161.

[9]. Koza, J.R., 1992, "Genetic Programming: On the Programming of Computers by Means of Natural Selection", MIT Press.

[10]. Aha, D.W., Kibler, D., and Albert, M. K., 1991, "Instance-based learning algorithms", Machine Learning journal, Vol. 6, No 1, Page(s):37-66.

[11]. Witten, I. H., and Frank, E., 2005, Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[12]. Quinlan, J.R., 1993, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Francisco, CA.

[13]. http://en.wikipedia.org/wiki/C4.5_algorithm last accessed on 11/11/14

[14]. http://www.uoguelph.ca/~qmahmoud/qws/dataset/ last accessed on 04/09/14

[15]. Al-Masri, E., and Mahmoud, Q. H., 2007, "Discovering the best web service", (poster) 16th International Conference on World Wide Web (WWW), 2007, pp. 1257-1258.

[16]. Al-Masri, E., and Mahmoud, Q. H., 2007, "QoS-based Discovery and Ranking of Web Services", IEEE 16th International Conference on Computer Communications and Networks (ICCCN), 2007, pp. 529-534.

[17]. Al-Masri, E., and Mahmoud, Q.H., 2008, "Investigating Web Services on the World Wide Web", 17thInternational Conference on World Wide Web(WWW), Beijing, April 2008, pp. 795-804. (for QWS WSDLs Dataset Version 1.0)

[18]. www.cs.waikato.ac.nz/ml/weka/ last accessed on 14/11/14