IJCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Vol.-6, Issue-11, Nov 2018 E-ISSN: 2347-2693

Improved Prediction Based Mining Approach for Classification using Association rules

Mittal.K[1*], Aggarwal.G[2] , Mahajan.P[3]

[1]Dept. of Computer Science, Jagannath University Delhi NCR, India ORCID:0000-0002-2967-0804
[2]Dept. of Computer Science, Jagannath University Delhi NCR, India
[3]Dept. of Computer Science , Institute of Innovation in Technology and Management, Guru Gobind Singh Indraprastha University, New Delhi India

[*]*Corresponding Author: kavitamittal.it@gmail.com, Tel.: 9873217562*

Available online at: www.ijcseonline.org

Accepted: 20/Nov/2018, Published: 30/Nov/2018

Abstract- Classification is one of the important data mining applications in the areas of decision sciences and knowledge extraction from the data. Classification using Association Rule Mining(ARM) is in great demand today with an aim of building moderate sized classifier consisting of limited number of rules from the database with higher classification accuracy rate. This classification approach integrates two important data mining strategies ARM and classification. Association rule mining aims to discover rules without any target based on association among the frequent items in the data where as classification based rule mining aims to discover targeted rules towards a predetermined class. The integration of these two techniques focuses on mining a set of Association Rules (CARs) which is a subset of association rules generate by some ARM technique. This integration also helps to resolve few problems associated with traditional classification systems. This paper attempts to improve the performance of CBA classifier with some modifications and performs the experimental evaluation against traditional classifier C4.5 in terms of error rate, number of classification association rules generated and the execution time.

Keywords- Rule mining, Classification Association rules, Classifier, Rule Pruning, CBA, Discretization.

I. INTRODUCTION

Classification is one of the important data mining applications in the areas of decision sciences and knowledge extraction from the data. Classification using Association Rule mining is in great demand today with an aim of building moderate sized classifier consisting of limited number of rules from the database with higher classification accuracy rate. This classification approach integrates two important data mining strategies. Association rule mining aims to discover rules without any target based on association among the frequent items in the data where as classification based rule mining aims to discover targeted rules towards a predetermined class. The integration of these two techniques focuses on mining a set of Association Rules (CARs) which is a subset of association rules generate by some ARM technique. This integration also helps to resolve few problems associated with traditional classification systems.

Classification based association rule mining focuses on producing a compact set of rules from the database to build moderate sized classifier with acceptable classification accuracy rate. The association rule mining approach initially generate all possible rules from the database satisfying the thresholds minsupp and minconf [1] with no specified target.ie the specified class. While classification with ARM aims to produce targeted rules i.e. the rules with a specific class attribute on their right side (consequent). This subset of special rules is called as classification Association Rules(CARs).In this chapter an existing classification based association rule mining technique with some modification is adopted to produce efficient CARs satisfying the minsupp and minconf thresholds. There is a need for adoption for the main reasons [2].

1) The algorithm is adjusted to work with transactional as well as non transactional database in order to exclude non targeted rules and mine only CARs .The aim is to reduce the number of rules generated and considering only the targeted rules.
2) Classification datasets may contain numeric and non numeric attributes. Mining of rules considering both data values becomes a difficult task [3].

The adoption process here requires preprocessing of data that involves discretization of attributes based on predetermined class or target[4]. The data mining process in the association classification model involves three stages:

1) Discretization of attributes
2) Discovering class association rules (CARs)
3) Building of classifier model using CARs

In Classification based on Association rules(CBA) algorithm , a heuristic approach is used to obtain a subset of classification rules in order to classify the training data set accurately . During classifier building process for preparing methods are used to remove contradicting and redundant rules and sometimes complicated pruning methods are likely to be used to enhance the accuracy of the classifier.

This study makes the following contributions:

1) It helps using ARM techniques for classification task.
2) It tries to solve the problems associated with traditional classification approaches.

This classification approach used in the study tries to resolve the problems related to classification as discussed in literature as follows:

1) The problem of understandability during the process of generating classification rules: Producing the rules satisfying minsupp amd minconf reduced the problem of biasness by generating only understandable rules.
2) Restricting the model to a limited number of rules by considering only significant rule based on some criteria like in our study only high confidence rules are allowed to participate in the classifier and eliminating less confidence rules resolves the problem of producing only interesting and useful rules in less time intervals.
3) In this technique, the database can be stored on the disk and instead of bringing the database in the main memory. The traditional classification systems works by loading the complete database in the main memory needing more memory space and CPU time. However sometimes there is a need for scaling the database to meet the memory and CPU needs [5].

Our study assumes the data in tabular form i.e. relational consisting of m cases with k attributes classified in n known classes. An attribute can be categorical or numeric. All the attributes are treated uniformly. We map each pair (attribute, integer value) pairs and a class label . Each pair (attribute, integer values) is called as item.

The dataset is represented by D and I represent the set of all the items in dataset D, and Y represents the set of class labels. A data case d belonging to D ($d \in D$) contains a subset of items X contained in I if X is contained in d. A CAR generated is of the form $X \rightarrow y$, where X is contained in I and $y \in Y$. A rule $X \rightarrow y$ is applicable to D with confidence c if D contains c% cases containing X with label y. The support of rule $X \rightarrow y$ is s if D contains s% cases containing X and label y. The objective to be achieved in this chapter is to :1) produce a set of CARs satisfying user defined thresholds. 2) to form a classifier from the CARs.

This paper is organized in the following manner: the section1introduces the concepts used in the paper, section2 presents the review conducted in support of execution of the algorithm, section3 describes the rule generation process with concept along with the procedure, section4 elaborates the classifier building approaches with different strategies with the procedure, section5 highlights the results based on the performance of different classification algorithm such as C4.5 the traditional classification algorithm and the proposed improved CBA algorithm with two versions M1 and M2 followed by their comparison. Finally the section6 concludes the findings.

II. RELATED WORK

The literature presents rich development in generation of association rules from large databases of different domains. There are different strategies implemented to improve the performances of the Apriori algorithm with additional features to overcome the issues related to number of useful rules discovered from the large databases . This section highlights the developments in regard to discovery of association rules as follows:

[1] presented an associative classifier that integrates the concept of classification and association rule mining to generate the set of CARs able to classify the datasets with high accuracy than other classifier discussed.

[2] tried to solve the problem of generating association rules from large transactional databases by presenting the new algorithm called Apriori Hybrid . This algorithm implements efficient scaling strategies and producing valuable rules.

[3] presents a discretization algorithm called CAIM Class-Attribute Interdependence Maximization for discretization of continuous attributes into discrete attributes to work with supervised learning algorithms.

[5] presented a novel association rule mining technique to discover association rules from large databases based on conventional Apriori approach with additional features in order to improve data mining performance.

[6] attempted to fix the mining problems in large databases by proposing an algorithm that incorporates buffer management, estimation and efficient pruning techniques.

[7] presented a new enhanced Apriori association rule mining approach that need less database scans and reduces the consumption of system resources with improved quality and efficiency.

[8] discussed the issue during building of scalable classifier and presents the decision tree classifier called SLIQ capable of handling both numeric and categorical attributes. It is based on breadth first tree growing strategy and also uses a new tree pruning algorithm to produce compact and accurate trees.

[9] presented a new associative classifier taking advantage of positive and negative association rules where negative association rules are rules that either associate negation of attribute values to classes or negatively associate attribute values to classes. It also present new way of pruning irrelevant classification rules using coefficient of correlation.

[10]explored ARM techniques to predict chronic diseases like diabetes etc as one of the application of ARM.

[11] used modified Apriori algorithm to mine the data from the cloud using association rule mining to overcome the major issue of multiple scan in the database for minimizing the space and time consumption.

III. GENERATING THE CARs

The study used AC algorithm CBA (Classification based on Associations) with improved version. It works in two steps [12]:1)generating rule using method CBA_Rule , 2) Building the classifier using method CBA_Classifier. CBA_Rule part generate rules based on Apriori methodology for finding rules. CBA_Classifier part participate in building the classifier.

A. Rule Generation concept

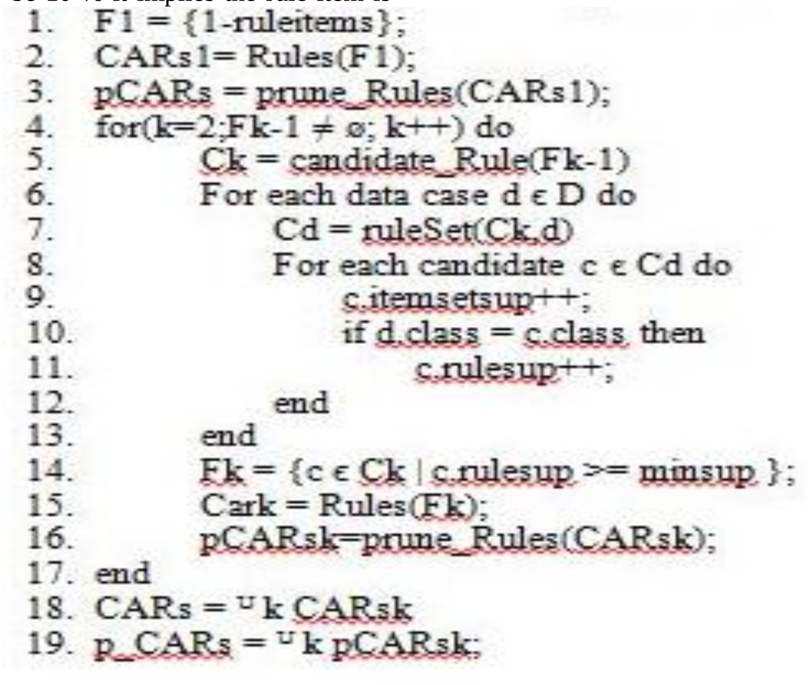
CBA_Rule is dedicated to discover the rule items crossing the minsup. The rule item takes the form <item set, y) where item set represent the set of items, y ϵ Y is the class label. The support of the item set (itemsetsup) represents the number of cases in dataset D containing the item set. The support of the rule item (rulesup) represents the number of case in ED containing the item set with class label y [7].

The support of rule item can be calculated as (rulesup/|D|)*100% where |D| represents number of cases in dataset and confidence can be calculated as (rulesup/itemsup)*100 %. The rule items satisfying minsup are considered as frequent rule items and rest as infrequent rule items. Suppose in a rule item < (x,1), (y,2)},class a}> where x, y are attribute. If support of item set I is 4 and support of rule is 3 and total number of training cases in D is 10, the support of rule item is 30% and confidence is 75%. If minsup is considered to be 10 % it implies the rule item is frequent as it satisfies the minsup threshold. Now among all the rule items with same itemsets , the rule item having highest confidence value is considered as applicable rule (AR) representing that particular set of rule items. If more than one rule item exist with same confidence value then the rule item with high support is selected randomly. If the confidence value of the rule is more than minconf then the rule is said to be accurate. If the support of the rule is higher than minsup the rule is said to be frequent. Therefore, the CARs generated is set of all applicable rules being accurate and frequent [6]

B. CBA_Rule Algorithm

The CBA_ Rule algorithm is responsible for generating all the rule items that are frequent by performing multiple iterations on data. Beginning with first pass, it calculates support of a particular rule item and indentify if it is frequent or not. Then in the next pass it begins with the initial set of the rule item that are frequent in the prior pass for generating new frequent rule items also called candidate rule items and soon[13]. At the end of each pass it tests if the rule is frequent by calculating its support during the pass. Then from the set of rule items that are frequent, it discovers CARs.

This section describes the process of generation of CARs based on the algorithm below in figure1. Let us consider k-rule item, a rule item with k-items, Fk,a set of frequent k-rule items where each rule item is of the form <(item set, itemstesup), (y,rulesupcount)>. Let Ck represent the set of candidate k-ruleitems. The CBA_Rule process works in the following manner:



```
F1 = {1-ruleitems};
CARs1= Rules(F1);
pCARs = prune_Rules(CARs1);
for(k=2;Fk-1 ≠ ø; k++) do
        Ck = candidate_Rule(Fk-1)
        For each data case d ϵ D do
              Cd = ruleSet(Ck,d)
              For each candidate c ϵ Cd do
                    c.itemsetsup++;
                    if d.class = c.class then
                          c.rulesup++;
              end
        end
        Fk = {c ϵ Ck | c.rulesup >= minsup };
        Cark = Rules(Fk);
        pCARsk=prune_Rules(CARsk);
end
CARs = ᵁ k CARsk
p_CARs = ᵁ k pCARsk;
```

Figure1: The CBA_Rules Algorithm

The lines 1-3 of the process in figure1 indicate the first pass that indentifies the frequent 1-rule items followed by generation of set of CARs using Rules method. CAR1 is obtained by using pruning operation that may be optional. The method prune_Rules applies pessimistic error based pruning method as used in C4.5 classification method, where pruning takes place in the following manner: The rule r with higher pessimistic error rate than the rule r', then the rule r will be pruned. This pruning method can reduce the number of rules generated. In each pass say kth pass, the process passes through major operations as follows:

1) The ruleitems Fk-1 discovered in (k-1)th pass participate in the generation of candidate rule items Ck using the method candidate_rule( ).
2) The algorithm produces the CARsk using the method Rules ( ).
3) The process is finalized by performing rule pruning on set of rules generated.

The functionality of the method candidate_Rule is similar to that of Apriori_Gen in Apriori algorithm. The method rule_Set uses the set of candidate rule items Ck and a datacase d to discover rule items from Ck, provided the data case d supports item sets of the rule items . The operation in lines 8-10 is similar to the above mentioned operation and also matches to Apriori algorithm.

In this process the support of itemsets and the ruleitems is updated separately where as in Apriori the count is updated only once. This count is also used during rule pruning process. Then finally the set of classification association rules (CARs) and the rule after pruning are stored in p_CARs.

IV. BUILDING THE CLASSIFIER (CBA_CB)

This process uses the modified CBA_CB algorithm to build the classifier using either CARs or p_CARs. The attempt to develop an efficient classifier from the complete set of rules need evaluation of all possible subsets of rules on the training dataset and coming up with the correct rule set with least error rate . To achieve the result there is a need to reduce the number of rules based on some criteria. The CBA algorithm being heuristic performs well as compared to C4.5, the traditional classification algorithm.

Prior to describing the process, let us consider the rule ranking criteria to be applied on the rules generated and selecting the rule to be the part of the classifier.
Given two rules ri and rj,

a) ri>rj, if confidence of ri is higher than that of rj
b) ri>rj, if confidence is same and ri has higher support than rj
c) ri>ri, if confidence and support are same and ri has less number of items in its antecedents than rj.
d) ri>rj, if confidence, support and cardinality are same and ri represents more classes than rj.

The rule ranking is needed to proceed for rule pruning aiming to enhance classification accuracy.
Given R, the set of generated CARs and let D the training dataset. The rule of algorithm is to select a set of high confidence rule from T to cover the dataset D.

The classifier follows the following structure:
<r1, r2,…,r n, class>, where ri є R, ra > rb if b>a, class is the class classifying the dataset.

For unseen data case, the first rule that satisfies the items in data case will classify it. If more than one rule classifies it then the rule with highest frequency will classify it. If no rule is applicable to the data case then it is assigned the default class.

A. The Modified Algorithm for CBA_CB:M1

The modified version M1 of the algorithm is proposed for building the classifier.
It consists of three steps:

Step1: Rank the set of generated rule in R according to the criteria mentioned above in order to ensure the selection of high confidence rule for the classifier.

Step2: Select the ranked rules .Then for each rule r in R, the dataset D is covered to identify the data cases that satisfy the rule items of the rule r. If the rule correctly classifies the data case d, it is marked. If r is able to classify at least one data case in D, then it is considered as one of the potential rule to be a part of the classifier. The data cases covered by the rule r are eliminated from D, when the rule selection is stopped for the classifier C then a default class will be selected for the remaining data that also becomes the default class of the classifier C. Then the total number of errors made by C can be computed by the sum of errors made by all selected rules in C and the default class in the training dataset. If no rule and no data case is left then the rule selection task is stopped and hence completed.

Steps 3- The rules not capable of improving the accuracy of the classifier are discarded. The first rule with least number of errors on D becomes the cutoff rule and all other rules after this that produce more errors are then discarded. The preserved rule and the default class from the classifier C can be obtained as follows in figure2:

```
1. R= sort ( R );
2. For every r є R in list do
```

```
3.          tmp = ø;
4.          for every case d ϵ D do
5.                if d matches the rule items of r then
6.                save d.id in tmp
7.                      and if r correctly classified then
8.                            mark r
9.                Check if r is marked then
10.                           Append r in C
11.                           Remove all data cases with some ids in tmp from D;
12.               Choose the default class for the C;
13.               Calculate the total number of error for C;
14.         End
15.   End
16.   Identify the first rule f in C with least number of errors and discard all other rules after the rule f.
17.   Mark the default class of p for C and return the classifier C.
```

Figure 2: Modified Algorithm for CBA_CB:M1

B. The Modified Algorithm for CBA_CB:M2

The algorithm takes care of two main constraints:

Constraint1: The rule with highest confidence and support among other rules cover the training case with an assurance that all training cases are covered. This can be achieved by the sorting criteria adopted.

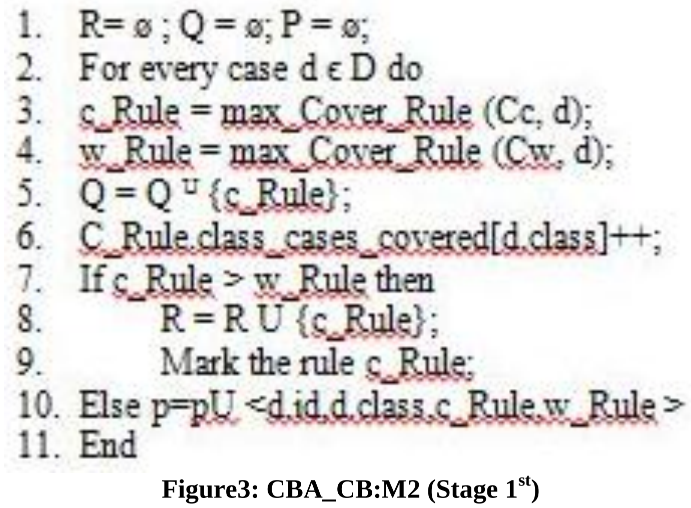
Constraint2: Each rule included in the classifier C when chosen must be able to correctly classify at least one of the remaining training cases in D.

The algorithm discussed is simple to implement and efficient if the database is small and resident in the main memory as it requires many passes on the database for generation of CARs.

For databases not resident in main memory i.e. for large databases the algorithm may show insufficient performance because of difficulties in database passes. To overcome this problem an improved version of the algorithm called CBA_CB:M2 is used that attempts to make one or two passes during the process of CARs generation. The concept it follows is that rather passing once cover the remaining data for every rule like in M1, it attempts to identify the best rule from R that can cover each case. M2 consists of three main stages [Liu,Hsu]:

Stage first: For every data case d, it finds the highest confidence rule (called c_Rule) that classifies d correctly and the highest confidence rule (called w_Rule) that classifies d wrongly [Antonie, Zaine] . If for some data case d, c_Rule > w_Rule then d must be classified by c_Rule. This procedure satisfies the above mentioned constraint1 and constraint2. Also the c_Rule that classifies the cases correctly is marked. If in case w_Rule > c_Rule then it becomes difficult to decide which rule will cover the case. In that case a data structure is formed of the form < d.id, y, c_Rule, w_Rule >, where d.id is unique identification number of the case d, y is the class of d.

Let P represent the collection of the forms < d.id, y, c_Rule, w_Rule >, Q is the set of all the c_Rules and R represents the c_Rules having higher confidence and than their corresponding w_Rule . The algorithm below in figure3 represents this stage of M2 process.



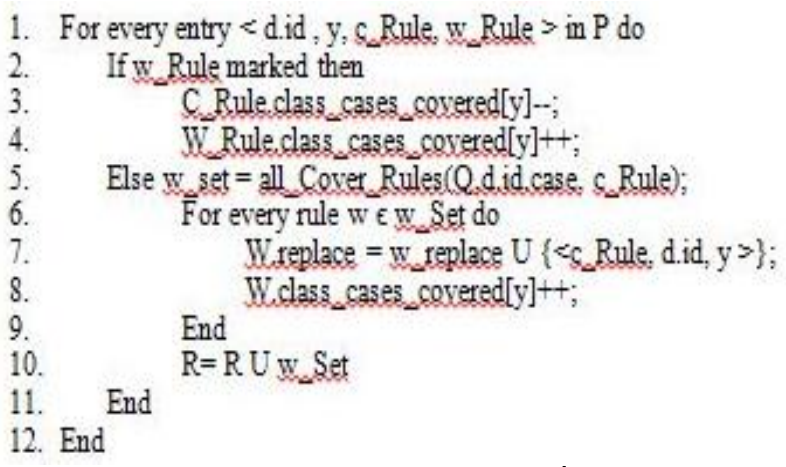
```
1.  R= ø ; Q = ø; P = ø;
2.  For every case d ϵ D do
3.  c_Rule = max_Cover_Rule (Cc, d);
4.  w_Rule = max_Cover_Rule (Cw, d);
5.  Q = Q ᵁ {c_Rule};
6.  C_Rule.class_cases_covered[d.class]++;
7.  If c_Rule > w_Rule then
8.          R = R U {c_Rule};
9.          Mark the rule c_Rule;
10. Else p=pU <d.id.d.class.c_Rule.w_Rule >
11. End
```

Figure3: CBA_CB:M2 (Stage 1st)

The max_Cover_Rule method discover high confidence rule that covers d. Cc and Cw are the set of rules that matches the class of d or not respectively. d.id and d.class indicate the unique identification number and the unique class of d respectively. For every c_Rule, it also counts the number of cases covered in each class using the field class_cases_covered of the rule.

Stage 2nd – For every case d if it is not decidable which rule will cover in stage1 , then d is tested again to identify the rules that classify d incorrectly and have high confidence with high support than the c_Rule corresponding to d. Thus this method may or may not make more than one pass over D.

The method below (figure4) describes the stage 2 of M2 as follows:



```
1.  For every entry < d.id , y, c_Rule, w_Rule > in P do
2.      If w_Rule marked then
3.          C_Rule.class_cases_covered[y]--;
4.          W_Rule.class_cases_covered[y]++;
5.      Else w_set = all_Cover_Rules(Q.d.id.case, c_Rule);
6.          For every rule w ϵ w_Set do
7.              W.replace = w_replace U {<c_Rule, d.id, y >};
8.              W.class_cases_covered[y]++;
9.          End
10.         R= R U w_Set
11.     End
12. End
```

Figure4: CBA_CB_M2 (stage 2nd)

If in the above process (figure4) w_Rule is marked it indicates that w_Rule cover the case represented by d.id. The function all_Cover_Rules find all the rules that classify the d.id wrongly and have higher confidence and support that its corresponding c_Rule. Such rules may replace c_Rule to cover the case d. This information is placed in the replace field of every rule (line7). The incremented value w.class_cases_correctly[y] indicate that the rule w_Rule may cover the case. R contains all the rules to be included in the classifier.

Stage 3rd – Its task is to select the set of CARs to build the classifier. It performs two steps shown in figure 5:

Step1: Select the set of sufficient rules to be included (line1-17) in the classifier. Initially R is ordered according to the criteria mentioned earlier to satisfy the constraint1 in the rule selection process. Line 1 and 2 of the algorithm represents initialization task. The function complete_class_Dist( ) counts the datacases of training dataset in each class as shown in line 1. The rule_Errors( ) method notify the number of errors made by all the selected rules on the training dataset.

```
1.  Class_Dist = complete_class_Dist
2.  Rule_Errors =0;
3.  R = sort (R);
4.  For every rule r in R in list do
5.      If r.class_cases_Covered[r.class] != 0 then
6.          For all < rule, d.id, y > in r.replace do
7.              If  the d.id case covered by p_r then
```

If the rule r do not classifies any training case correctly, it is discarded as shown in line 5. Else r will be included in the classifier. This process satisfies the constraint 2. R tries to replace all the rules in r.replace if r proceeds them as shown by line 6 but with the condition that id d.id case is already covered by the previous rule then current rule r will replace the rule to cover the data case and the class_cases_Covered fields of r and previous rule p_r will then be updated accordingly as shown in line 7-9.

For every rule selected rule_Errors and class_Dist fields are uploaded as in line 10-11. Also a default class, default _class is selected, based on the existence of frequent class in the remaining training datacases, and computed based on the value of class_Distr. Once the default class is selected, its number of errors called default_Error is also computed based on the errors made by the default class in the remaining training datacases as in line 13. Then total_Errors representing the total number of errors made by selected rules in the classifier c and its default class can be computed as rule_Errors + default_Errors as shown by line 15.

Step2: The rules introducing more errors are discarded in line 18-20 and the final classifier C is returned

```
8.                    r.class_cases_covered[y]--;
9.                Else
10.                   P_r.class_cases_covered[y]--;
11.               Rule_Errors = rule_Errors + errors_of_Rule(r);
12.               class_Dist = update (r,class_Dist);
13.               default_class = select_Default (class_Dist);
14.               default_Errors = def_Errors (default_Class, class_Dist);
15.               total_Error = rule_Errors + default_Errors;
16.               Insert < r, default_Class, total_Errors >at the end of C
17.           End
18.   End
19.   Find the first rule p in C with lowest total_Errors and discard all other rules after p from c;
20.   Add default class of p to the end of c;
21.   Return the classifier C with total_Errors and default_class;
```

Figure 5: CBA_CB:M2 (stage 3rd)

V. EMPIRICAL EVALUATION

This section compares the classifier produced by the algorithm CBA with variants M1 and M2 with C4.5 (based on tree and rules). We cover 10 datasets from UCI machine learning repository [14] for the analysis. The analysis show the performances of the algorithms based on execution time, number of rules and error rate.

The experiment is performed keeping minconf value 50% but deciding minsup and minconf is a bit complex task as it has significant impact of the quality and efficiency of the classifier. If minsup value is kept high than the possible with high confidence then the rules may not satisfy the minsup value and will not be included in the classifier and hence the CARs may not be able to cover the training data sets and accuracy may be reduced. If support value lies between 1-2% then the CARs generated are capable of building the accurate classifier. In our experiment we kept the minsup value 2% as keeping the minsup value 1% may include insignificant rules that may reduce the accuracy of the classifier. Moreover large number of rules will become unmanageable and difficult to make accurate predictions. Discretization is used for continuous attribute to convert into required format. The data is taken from machine learning library. The experiment began by setting all the parameters in C4.5 with their default values. The error rates are calculated on each and dataset from 10-fold cross validation. The results are presented in the table1 below:

Table 1: Experimental result comparing the error rate

S.No	Datasets	C4.5 Error rate (%)	CBA M1 Error rate (%)	CBA M2 Error rate (%)
1	Iris	7.2	0.0	0.0
2	Car Evaluation	11.4	3.2	0.0
3	Banknote Authentication	9.2	0.3	0.0
4	Facebook	18.6	7.6	2.4
5	Forest	9.6	1.9	2.8
6	Horse Colic	17.9	0.3	0.3
7	Indian Liver patient	8.2	0.2	2.2
8	Monks	7.1	0.0	0.0
9	Wine	8.4	0.6	0.6
10	Vertebra	7.4	0.0	0.0
	Average error rate	10.5	1.41	0.83

Column1: It display the names of the datasets used for evaluation of the algorithms.
Column2: It presents C4.5 rules mean error rates over the 10 fold cross validation using discretized data sets.
Column3: It gives average error rates using CARs in our classifier developed using CBA with minsup 2% and minconf 50% over 10 fold cross validation pruning with M1.
Column4: It gives average error rates using CARs in our classifier developed using CBA with minsup 2% and minconf 50% over 10 fold cross validation pruning with M2.

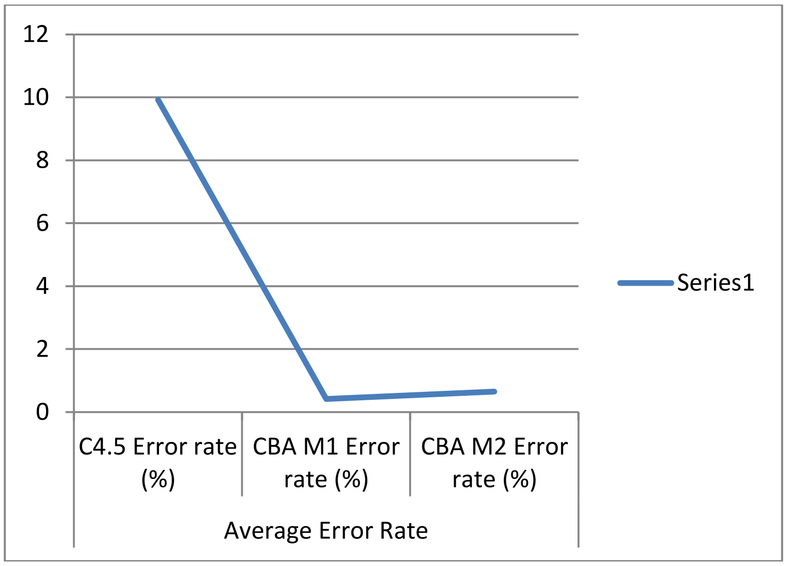


Figure 6: Average Error Rate

It can be depicted from the above figure6 results that CBA produces more accurate classifier than C4.5 rules where the average error rate in C4.5 is 10.5 that decreases to 1.41 for CBA. Furthermore, our classification system is superior to C4.5 rules, the traditional classification method on almost every dataset. It is observed that without or with rule pruning the accuracy of the obtained classifier does not differ to a major extent, they are almost same. Thus, the rules produced with p_CARs (after pruning) are capable of building the accurate classifier. Thus, comparing in this respect CBA is more efficient than C4.5.

The table2 below presents the number of rules obtained:

Table2: Experimental results showing number of Rules

S.No	Datasets	C4.5 Number of Rules	CBA M1 Number of Rules		CBA M2 Number of Rules	
			With Pruning	Average Rules in classifier	With Pruning	Average Rules in classifier
1	Iris	7	43	9	58	19
2	Car Evaluation	18	150	21	58	19
3	Banknote Authentication	12	58	24	58	19
4	Facebook	36	141	27	142	1
5	Forest	40	177	30	179	13
6	Horse Colic	32	242	33	243	1
7	Indian Liver patient	48	231	34	230	1
8	Monks	33	227	21	231	12
9	Wine	8	181	25	179	18
10	Vertebra	23	117	14	117	6
	Average	26	157	24	150	11

Column1: It display the names of the datasets used for evaluation of the algorithms.
Column2: It presents the average number of rules produced by C4.5 based on 10 fold cross validation.
Column3: It gives the average number of rules with CBA M1. The first part shows the number of rules produced after M1 pruning and second part displays the average number of rules in the classifier built by CBA CB M1 using p_CARs..
Column4: It shows the average number of rules with CBA M2. The first part shows the number of rules produced after M2 pruning and second part displays the average number of rules in the classifier built by CBA CB M2 using p_CARs..

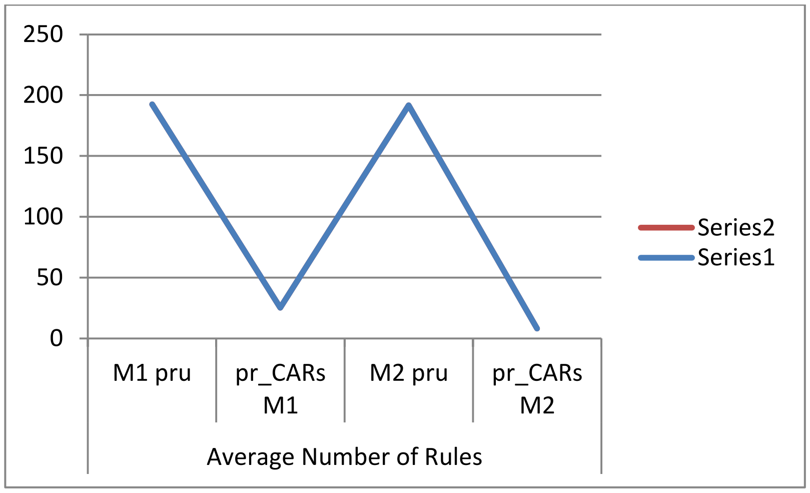


Figure 7: Average Number of Rules

The study shows the average number of rules shown in figure7 in the classifier built by CBA-CB using p_CARs, are more in CBA classifier as compared to C4.5 in few cases as the aim is to cover unseen data cases with highest accuracy. However CBA-CB generates more understandable and efficient rules that may be generated by C4.5 as it randomizes the selection of attributes in case of more number of attributes. Also CBA CB M1 presents more understandable and efficient rules than CBA CB M2 as in some cases the rules produced are very less that they may not be able to cover all the data cases.

The table3 below presents the execution time during generation of rules:

Table3: Experimental results showing the execution time in generation of Rules

S.No	**Datasets**	**C4.5 Execution time(s)**	**CBA M1 Execution time RG(pr) p_CARs**	**CBA M2 Execution time RG(pr) p_CARs**
1	Iris	0.0383	0.15 0.00	1.19 0.07
2	Car Evaluation	0.0817	1.85 0.05	1.19 0.07
3	Banknote Authentication	10.9082	1.22 0.04	1.19 0.07
4	Facebook	5.44501	1.86 0.03	1.76 0.04
5	Forest	4.717	5.45 0.02	5.49 0.05
6	Horse Colic	0.58929	5.51 0.03	5.44 0.05
7	Indian Liver patient	1.436	11.18 0.06	11.19 0.09
8	Monks	0.02255	1.58 0.01	1.48 0.05
9	Wine	0.39680	3.74 0.00	3.81 0.02
10	Vertebra	0.908	6.21 0.01	6.58 0.03
	Average	2.45	3.86 0.025	3.93 0.054

Column1: It display the names of the datasets used for evaluation of the algorithms as mentioned earlier.
Column2: It presents the average time consumed in generation of rules produced by C4.5 based on 10 fold cross validation.
Column3: It presents the average time consumed in generation of rules produced RG and by CBA M1 using p_CARs with pruning.
Column4: It presents the average time consumed in generation of rules produced by RG and CBA M2 using p_CARs with pruning.

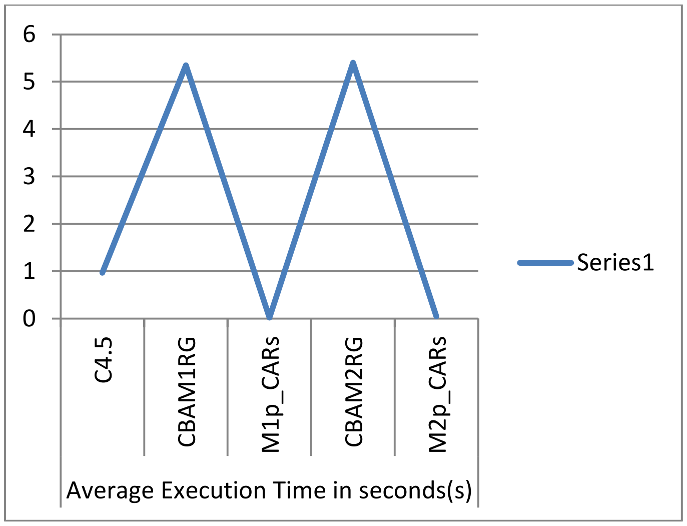


Figure 8: Average Time Execution in seconds (s)

In terms of time consumption M1 is more efficient than M2 as shown in figure8.

VI. CONCLUSION

This chapter presents the improved predictive mining approach integrating the classification and association rule mining in the algorithm CBA-CB. The algorithm presented generates all classification association rules (CARs) for developing the classifier with highest accuracy using enhanced ranking and pruning method. The classification approach used tries to fix the problems related to classification discussed in literature such as number of rules, time consumption, memory usage, maintenance and understanding of the rules produced, accuracy of the classifier. The classifier presented attempts to meet the above mentioned issues in the classification. The results indicate the improved performance of CBA_CB:M1 classifier as compared to CBA_CB:M2 that differ in their pruning approach and the traditional classifier C4.5 in terms of accuracy, average number of rules and time consumption. The will extend the experiment with CBA_CB:M1 classifier in the future works. However in the next study some more issues related to the coverage of database will be discussed and attempt will be done to fix these issues.

REFERENCES

[1]. Liu, B., Hsu, W., Ma, Y. CBA: Integrating Classification and Association Rule Mining. In KDD'98, New York, NY, Aug. 1998.

[2]. Agrawal,R. and Srikant, R., Fast algorithms for mining association rules, in Proc. 20th Int. Conf. Very Large Data Bases, VLDB, edited by J.B. Bocca, M. Jarke, and C. Zaniolo, Morgan Kaufmann 12 (1994) 487-499.

[3]. Kurgan. L and Cios, K.J.: Discretization Algorithm that Uses Class-Attribute Interdependence Maximization, Proceedings of the 2001 International Conference on Artificial Intelligence (IC-AI 2001), pp.980-987, Las Vegas, Nevada. (Pub. 2001.)

[4]. PDF) Data Mining Discretization Methods and Performances. Available from: https://www.researchgate.net/publication/264886861_Data_Mining_Discretization_Methods_and_Performances [accessed Jul 03 2018].

[5]. AL-Zawaidah, Farah Hanna., Jbara ,Yosef Hasan, "An Improved Algorithm for Mining Association Rules in Large Databases", World of Computer Science and Information Technology Journal (WCSIT), ISSN: 2221-0741 Vol. 1, No. 7, 311-316, 2011. 7.

[6]. Agrawal R., Imielinski T. and Swami A. (1993) Mining Association rules between sets of items in large databases, In the Proc. of the ACM SIGMOD International Conf. on Management of Data (ACM SIGMOD , 93), Washington, USA, 207-216.

[7]. Hassan M. Najadat, Mohammed Al-Maolegi, Bassam Arkok, "An Improved Apriori Algorithm for Association Rules", International Research Journal of Computer Science and Application Vol. 1, No. 1, June 2013, PP: 01 – 08.

[8]. Mahta, M., Agrawal, R. and Rissanen, J. 1996. SLIQ: A fast scalable classifier for data mining. Proc. of the fifth Int'l Conference on Extending Database Technology.

[9]. Antonie, M. . Za¨ıane ,O. R. An Associative Classifier based on Positive and Negative Rules. In DMKD '04 Paris, France, June 13, 2004,

[10]. S.Rani, S.Kaushik, "Application of Data Mining techniques for predicting Diabetes" in International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Vol 3(3),pp-1996-2004, 2018.ISSN-2456-3307.

[11]. A.Sharma, N.K.Tiwari. " Mining Association Rules in Cloud Computing Environments using Modified Apriori" in International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Vol 3(1),pp-629-634, 2018.ISSN-2456-3307.

[12]. http://cgi.csc.liv.ac.uk/~frans/KDD/Software/CBA/cba.html

[13]. ANALYTICS VIDHYA CONTENT TEAM, Mining frequent items bought together using Apriori Algorithm (with code in R) AUGUST 11, 2017 https://www.analyticsvidhya.com/blog/2017/08/mining-frequent-items-using-apriori-algorithm/

[14]. Merz, C. J. & Murphy, P. (1996). UCI Repository of Machine Learning Database. Available: http://www. ics.uci.edu/~mlearn/MLRepository

Author's Profile

Ms. Kavita Mittal is a research scholar and Assistant Professor at the Department of Computer Science and Engineering, at the Jagannath University, Haryana, India. She has 13 years of teaching experience in higher education and has published around 15 research papers in the field of Data Mining , Big Data Analysis , Business Intelligence etc.

Dr. Gaurav Aggarwal is a research guide and Associate Professor and Head at the Department of Computer Science and Engineering, at the Jagannath University, Haryana, India. He has 13 years of teaching experience in higher education and has published around 15 research papers in the field of Data Mining , Software Reliability, Networking etc.

Dr. Prerna Mahajan is a research guide and Professor and Head at the Department of Computer Science , at the Institute of Information Technology and Management affiliated by Guru Gobind Singh Indraprastha University, New Delhi, India. She has 13 years of teaching experience in higher education and has published around 18 research papers in the field of Data Mining , Big Data Analysis , Business Intelligence etc.