Association Rule Classification and Regression Algorithm Based on Frequent Itemset Tree

Ling Wang*, Hui Zhu and Ruixia Huang
Key Laboratory of Knowledge Automation for Industrial Processes of Ministry of Education, School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China
*Corresponding author

***Abstract*—The categorization association rules based on the Apriori algorithm can't deal with the numerical data directly. When mass rules are generated, classifying the new data enjoys matching so many rules one by one as to decrease the efficiency and accuracy. Moreover, the association rules can't be used to realize the regression prediction. In order to solve above problems, we proposed a new association rule classification and regression algorithm based on frequent itemset tree (ARCRFI-tree) according to the advantages of matrix operation and tree structure. Firstly, all frequent itemsets are obtained by constructing a new frequent tree structure, based on which the association rules are mined. Then, the consequents of the association rules are reconstructed with the least square method to realize the classification and regression prediction for new sample. Finally, the theoretical analysis and experiments compared with algorithms demonstrate our algorithm has high prediction accuracy and mining efficiency.**

Keywords*—matrix operation; frequent itemset tree; association rule; classification; regression

I. Introduction

Data mining is an emerging technique, whose goal is to extract significant patterns or interesting rules from transaction databases. In recent years, it has been successfully applied to many fields, including user behavior analysis, network intrusion detection, event classification and regression, etc. Association Rule is a typical data mining method, which is used to find the potentially meaningful correlations among data itemsets. It can be decomposed into two subproblems: one is to find all frequent itemsets which meets with the minimum support threshold and the other is to generate all association rules which satisfy the minimum confidence threshold.

The Apriori algorithm was originally presented by Agrawal and Srikant [1], which can be used to discover the interesting relationships between itemsets. But there are still some problems such as repeated scanning of transaction database, too many candidate itemsets generated and lower efficiency. The improvements of traditional association rule mining are mostly based on it. In [2-5], the transaction database is transformed into the Boolean matrix. Though the transaction database only need to be scanned once based on matrix operation, and the efficiency of algorithm is improved, there are still many candidate items generated. The FP-growth algorithm[6], proposed by Han et al., aimed to avoid generating candidates and enhance the efficiency by constructing the frequent pattern tree (FP-tree). However, it still needs to scan the transaction database twice, which is not suitable for massive transaction data mining.

The current association rules have been applied to classification according to the correlation between the frequent itemsets and the class label. CBA algorithm [7] aims to achieve the classification based on the mining rules which are obtained by the Apriori algorithm. Though the classification for new samples can be realized, it has some drawbacks, such as producing lots of candidates, repeating to scan transaction database and low mining efficiency. FP-growth algorithm-based CMAR algorithm [8] constructs the CR-tree to store the rules and scans the transaction database only twice, but it still needs a large amount of memory. Nevertheless, the practitioners and researchers hope to explore these rules to predict future behavior by estimating values of certain attributes based on some other. So far, Association Rule is not used to solve such estimation problems by mining the relationship between attributes.

To solve the problem aforementioned, in this paper, combined with matrix operation and tree structure, an Association Rule Classification and Regression Algorithm based on frequent itemset Tree (ARCRFI-tree) is presented. The transaction data is discretized and the frequent itemsets are generated according to the matrix operation, which avoids repeated scanning for the database. Considering the advantage that the tree structure does not need to generate candidates in rule mining, a new frequent tree structure is proposed. Combined with the existing rules and the least square method, the consequents of the rules are obtained to realize the classification and regression prediction for the new sample data.

II. Basic Conception

Definition 1 Properties of discretization representation.

Let $D=\{T_1,T_2,\cdots,T_N\}$ be original dataset, where N is the number of data samples. Each sample $T_i=\{x_1^i,x_2^i,\cdots,x_n^i\}$ is associated with a unique identifier. $x_j^i(1\le i\le N,1\le j\le n)$ represents the j th quantitative value of the i th transaction. Based on the discretization with clustering algorithm, $\boldsymbol{x}_j$ is discretized as itemsets $I_j=\{I_{j,1},I_{j,2},\cdots,I_{j,n_j}\}$, where I_{j,n_j} denotes the n_j th discrete interval $[\min(I_{j,n_j}),\max(I_{j,n_j})]$.

Definition 2 Boolean matrix M .

TABLE I. BOOLEAN MATRIX TRANSFORMATION

Discrete transaction database D_Z

Transaction	Item Set
T_1	I_1, I_3, I_4
T_2	I_2, I_3, I_5, I_6
T_3	I_1, I_5
T_4	I_3, I_4, I_6

$\longrightarrow$

Boolean matrix M

Item / Transaction	I_1	I_2	I_3	I_4	I_5	I_6
T_1	1	0	1	1	0	0
T_2	0	1	1	0	1	1
T_3	1	0	0	0	1	0
T_4	0	0	1	1	0	1

The transaction database D are discretized as $D_Z = \{T_1', T_2', \cdots, T_N'\}$. $I = \{I_1, I_2, \cdots, I_m\}$ is a set of items, T_i' is a set of different items of itemsets and $T_i' \in I (1 \le i \le N)$. Boolean matrix M is the conversion representation of discrete transaction database D_Z . Let's assumed that the discrete transaction database contains N discrete transactions, and each of them contains m items. Then D_Z is converted to a Boolean matrix M . Suppose there are N discrete transactions contained in discrete transaction database and m items included in each discrete transaction. Then,

$$f : D_Z \to M \tag{1}$$

Where, $$M = f(D_Z) = (M_{ij})_{N \times m}, \tag{2}$$

$$M_{ij} = \begin{cases} 1 & I_j \in T_i \\ 0 & I_j \notin T_i \end{cases} (i = 1, 2, \cdots, N; j = 1, 2, \cdots, m) \tag{3}$$

That is, if I_j is included in T_i , the element of the j th column and the i th row in M is 1 , otherwise is 0. Items contained in the discrete transaction database D_Z are converted to column vectors, and each discrete transaction is converted to the row vector (TABLE I).

Definition 3 Frequent itemset.

Itemset L_k can be called frequent itemset if the support $supp(L_k)$ is greater than $\min supp$ or equal to it, where $\min supp$ is the minimum support threshold.

Definition 4 Support.

Let D be transaction database, D_Z be the corresponding discrete one, and M be the converted Boolean matrix. The support $supp(L_k)$ of frequent k -itemset $L_k = \{I_1, I_2, \cdots, I_k\}$ can be calculated by the element-wise product of the corresponding column vectors of the I_k in the Boolean matrix M . As we can see from TABLE I, the 3-itemset $L_3 = \{I_1, I_3, I_4\}$, the column vectors corresponding to I_1, I_3, I_4 are $(1,0,1,0)^T, (1,1,0,1)^T, (1,0,0,1)^T$ respectively, and $supp(L_3) = 1$.

Definition 5 Frequent itemset tree.

The frequent itemset tree is used to store the frequent itemsets and corresponding position information generated in the process of mining association rules, which can be described as follows: (1) It is made up of root node $Root$ and child node $node_c$. (2) Each child node $node_c$ comprises two data fields, itemset and column vector. In which, itemset refers to the set of items stored in the node, and column vector refers to the position information of the itemset in the discrete transaction database, which is used to construct the tree structure. (3)The order of searching tree structure is a top-down way, and only the first and second layers need to be constructed. After that, the whole frequent tree can be automatically generated based on the existing information. Finally, all frequent itemsets and corresponding support can be obtained by traversing the each node of tree structures. (4) The transaction database only needs to be scanned one time without generating candidates. Besides, the construction of tree structure and frequent itemsets mining are performed simultaneously, which improves the mining efficiency of association rules.

III. Association Rules Classification and Regression Algorithm Based on Frequent Itemset Tree (Arcrfi-Fit)

In this paper, a frequent itemset tree-based association rules classification and regression algorithm is proposed, which combines the advantages of the matrix operation and tree structure. Firstly, the transaction database is discretized based on the DFAC algorithm [9]. Then the discrete transaction database is converted to a Boolean matrix. Frequent itemset tree is constructed to get all frequent itemsets. With these itemsets, association rules are generated. Combined with the least square method, the consequents of the association rules can be constructed to realize the classification and regression prediction for new sample data.

A. *Association Rules Mining Based on Frequent Itemset Tree*

Compared to the existing association rules mining algorithms, the frequent itemset tree-based association rules mining algorithm only needs to scan the database once, and only the first two layers of tree structure need to be constructed. The whole tree structure can be automatically obtained by the information stored in the tree node. Furthermore, all the frequent itemsets are obtained, which improves the mining efficiency.

The basic flow of the algorithm is described as follows:

Input: $min\,supp$, $min\,conf$, transaction database $D=\{T_1,T_2,\cdots,T_N\}$

Output: Association Rules ($Rule$)

Step 1 The discretization is realized with DFAC clustering method, and the discrete data D_Z is converted to Boolean matrix M .

Step 2 Frequent 1-itemset is generated based on the Boolean matrix M . The sum of column vectors v_{L_1} in the Boolean matrix M is denoted by $supp(L_1)$. Compared with the predefined minimum support ($minsupp$), item L_1 is called a frequent 1-itemset when $supp(L_1)\ge minsupp$ .Then the item and its corresponding column vectors $<L_1,v_{L_1}>$ would be stored in the frequent 1-itemset information set $S_1=\{<L_1,v_{L_1}>_1,<L_1,v_{L_1}>_2,\cdots,<L_1,v_{L_1}>_l\}$. If $supp(L_1)<min\,supp$, the item L_1 is not a frequent 1-itemset.

Step 3 The initial frequent set tree are constructed with the frequent 1-itemset information set $S_1=\{<L_1,v_{L_1}>_1,<L_1,v_{L_1}>_2,\cdots,<L_1,v_{L_1}>_l\}$.

Step 3.1 Create the root node $Null$ of the tree, based on which the new child nodes are constructed. According to the left-to-right order, the frequent 1-itemset and corresponding column vectors $<L_1,v_{L_1}>(1\le i\le l)$ are sequentially added into the first level child nodes of the tree structure.

Step 3.2 Connect the two itemsets from different attributes of the first level child nodes. Suppose that the items contained in the two child nodes are I_{i_1,j_1} and $I_{i_2,j_2}(i_1,i_2=1,2,\cdots,n;i_1\ne i_2)$, we can get the 2-itemset $\{I_{i_1,j_1},I_{i_2,j_2}\}$, in which the corresponding column vectors are $v_{I_{i_1,j_1}}$ and $v_{I_{i_2,j_2}}$ respectively. $supp(\{I_{i_1,j_1},I_{i_2,j_2}\})$ can be obtained based on the inner product operation of $v_{I_{i_1,j_1}}$ and $v_{I_{i_2,j_2}}$. If $supp(\{I_{i_1,j_1},I_{i_2,j_2}\})\ge minsupp$, the child node including the item I_{i_1,j_1} can be viewed as a parent node to construct the new child nodes, and the data field is $<\{I_{i_1,j_1},I_{i_2,j_2}\},v_{\{I_{i_1,j_1},I_{i_2,j_2}\}}>$, where $v_{\{I_{i_1,j_1},I_{i_2,j_2}\}}=v_{I_{i_1,j_1}}\cap v_{I_{i_2,j_2}}$.

Step 4 Construct the whole frequent itemset tree automatically with the initial frequent itemset tree.

Step 4.1 Suppose there are $k(k\ge 2)$ layers in the current tree and l leaf nodes $\{node_k^1,\cdots,node_k^i,\cdots,node_k^l\}(1\le i\le l)$ in the k th layer. Where $node_k^i$ represents the i th leaf node in the k -th layer and it includes the data domain $<\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k}\}>,v_{\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k}\}}$.

Step 4.2 For the leaf node $node_k^i$, find the child node including the data fields $<\{I_{node_k^i,k},*\},v_{\{I_{node_k^i,k},*\}}>$ in the second layer, where $*$ denotes the arbitrary elements. Find the child node which contains the data fields $<\{I_{node_k^i,k},*\},v_{\{I_{node_k^i,k},*\}}>$ in the first layer, then $supp(\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k},*\})$ can be computed based on the element-wise product between vectors $v_{\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k}\}}$ and v_* . If $supp(\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k},*\})\ge minsupp$, the $node_k^i$ is chosen as the parent node to construct the new child nodes which contains the data domains $<\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k},*\},v_{\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k},*\}}>$. Otherwise, the next child node is found and determined. Here $v_{\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k},*\}}=v_{\{I_{node_k^i,1},I_{node_k^i,2},\cdots,I_{node_k^i,k}\}}\cap v_*$.

Step 4.3 Repeat step 4.2 until the leaf nodes are traversed or there is no frequent itemset to generate.

Step 5 Get all frequent k -itemsets F_k and generate the association rules $Rule$. The k th child node of the frequent itemset tree is traversed in turn. The itemset of the data field is frequent k -itemset F_k and the sum of the column vectors is the support $supp(F_k)$. The association rules are generated and output based on the frequent itemsets and minimum confidence ($minconf$).

B. Classification and Regression Algorithm Based on Association Rules

In order to improve the prediction accuracy of association rules, the classification and regression association rules are firstly reconstructed by combining the least square method with the extracted rules, and then the new samples are matched with the rules efficiently, on this basis, the best rule is directly identified to match the new sample and predict its output.

The specific process is as follows:

Input: Association rules $Rule$, new samples $T=\{x_1,x_2,\cdots,x_{n-1}\}$

Output: The output category and specific value of the new sample data.

Step 1 Prune the rules ($Rule$).

For the $Rule1$ and $Rule2$, if the antecedents of them exist inclusion relation and the consequents are identical, there are some redundancies need to be removed. Furthermore, if $Rule1$ satisfies one of the following conditions, $Rule2$ is required to be pruned.

1) $supp(Rule1) > supp(Rule2)$

2) $supp(Rule1) = supp(Rule2), conf(Rule1) > conf(Rule2)$

3) $supp(Rule1) = supp(Rule2), conf(Rule1) = conf(Rule2)$, and the itemsets contained in the antecedent of the $Rule1$ is less than that of $Rule2$.

Step 2 Construct the consequents of the rules.

The rules obtained by pruning are matched with the sample in the discrete transaction database D_Z to find all identifiers of samples that match with the rules. And then the transactions corresponding to sample transaction identifiers are taken out from the transaction database D . The least square method is applied to get the consequent. The obtained rule is described as:

$$\begin{aligned} Rule: If\ \ & x_1 \in [min(I_{1,n_1}), max(I_{1,n_1})], \cdots, \\ & x_j \in [min(I_{j,n_j}), max(I_{j,n_j})], \cdots, \\ & x_{n-1} \in [min(I_{n-1,n_{n-1}}), max(I_{n-1,n_{n-1}})], \\ Then\ \ & x_n \in [min(I_{n,n_n}), max(I_{n,n_n})], \\ & x_n = a_0 + \cdots + a_j x_j + \cdots + a_{n-1} x_{n-1} \end{aligned} \tag{4}$$

Where, $I_{j,n_j} (1 \le j \le n-1)$ denotes the n_j th discrete interval of the x_j , $min(I_{j,n_j})$ and $max(I_{j,n_j})$ denote the minimum and maximum of the discrete interval I_{j,n_j} , a_j denotes the coefficient of the consequent.

Step 3 Classification and regression prediction for the new sample.

First, discretize the new sample $T = \{x_1, x_2, \cdots, x_{n-1}\}$. Next is to determine whether there are rules matched with the data. If the rules exist, the category of the new sample is identical with the rule of the highest confidence, and the output can be obtained based on the consequent of the rule. If the rules do not exist, the output can be predicted according to the fuzzy inference [10]:

$$y = \sum_{l=1}^{r} w_l y^l \Big/ \sum_{l=1}^{r} w_l \tag{5}$$

Where, r denotes the number of association rules, y^l denotes the quantitative output of the $l (1 \le l \le r)$ th rule, y denotes the quantitative output, $w_l = \prod_{i=1}^{n} \mu_i^l$, μ_i^l denotes the membership degree of the $i (1 \le i \le n)$ th itemset in the l th rule for the discrete transaction(as shown in equation (6)).

$$\mu_i^l = e^{-\frac{(x_i - \overline{x}_i^l)^2}{2(\sigma_i^l)^2}} \tag{6}$$

Where, $\overline{x}_i^l$ and σ_i^l represent the mean and the standard deviation of the i th itemset in the l th rule respectively.

IV. Algorithm Analysis and Simulation Experiments

To test the performance and effectiveness of the proposed algorithm, the UCI benchmark databases (Iris, Wine, Seeds, Housing) [11] and rolling dataset (as shown in TABLE II) are selected. Compared with the Apriori and FP-growth algorithm, the ARCRFI-tree algorithm is analyzed in complexity, running time and classification or prediction accuracy.

TABLE II. DATASETS USED IN THE EXPERIMENTS

Dataset	Attributes	Instances
Iris	4	150
Wine	13	178
Seeds	7	210
Housing	14	506
Rolling	23	1500

Experimental environment is Intel Pentium CPU, 6 GB memory, 500 GB hard disk, Windows 7 flagship operating system, and the software used is MATLAB R2014a.

A. *Complexity Analysis*

For the transaction database $D = \{T_1, T_2, \cdots, T_N\}$ of length N , $T_i = \{x_1^i, x_2^i, \cdots, x_n^i\}$ denotes the i th transaction, $x_j^i (1 \le i \le N, 1 \le j \le n)$ denotes the value of the j th attribute in the i th transaction. Suppose $|F_k|$ denotes the number of frequent k -itemsets and $|C_k|$ denotes the number of candidate k -itemsets. In worst case, each data attribute will be divided into N discrete intervals. Under the above conditions, to test the performance of the proposed ARCRFI-tree algorithm, the four algorithms are compared in terms of scanning time, time complexity and space complexity of the original database in the process of mining association rules from the perspective of theoretical analysis.

From the TABLE III, the ARCRFI-tree algorithm proposed in this paper needs to scan the original database only once, which is more efficient than other algorithms. From the time complexity, if the size of transaction database is small, the difference between FP-growth and ARCRFI-tree will be not obvious. With the increasing of transaction database, the time complexity of ARCRFI-tree algorithm is significantly smaller than that of others. We can see that the space complexity of the proposed ARCRFI-tree algorithm is slightly higher than that of the FP-growth, which is because the ARCRFI-tree needs to convert the discrete transactions into Boolean matrix, but it is still better than Apriori algorithm. Therefore, compared with the other algorithms, we can draw a conclusion that the proposed algorithm has a better performance.

TABLE III. THE COMPARISON FOR COMPLEXITY

Algorithm	Scan Times	Time Complexity	Space Complexity
Apriori	k	$O(n\times N^2+\sum_{k\geq 2}\lvert L_k\rvert\times\lvert L_k\rvert+\lvert C_k\rvert\times(N+1))$	$O(n\times N+\lvert L_k\rvert+\lvert C_k\rvert)$
FP-growth	2	$O(n\times N\times(3(N+1)/2))$	$O(2\times(n\times N)+\lvert L_k\rvert)$
ARCRFI-tree	1	$O((n\times N)\times(1+k))$	$O((1+N)\times(n\times N)+\lvert L_k\rvert)$

B. *Performance Comparison with Different Algorithms*

To verify the computational performance of ARCRFI-tree algorithm, in this part, with the datasets of Iris, Wine, Seeds and Housing, the running time and the number of rules of each algorithm is analyzed under the different minimum support. The experimental results are shown in Figure 1 and Figure 2.

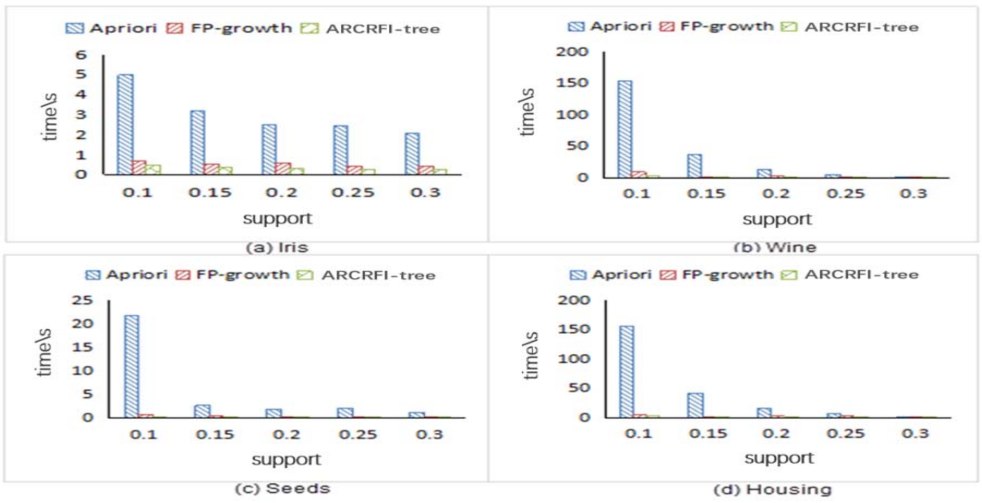

FIGURE I. RUNNING TIME

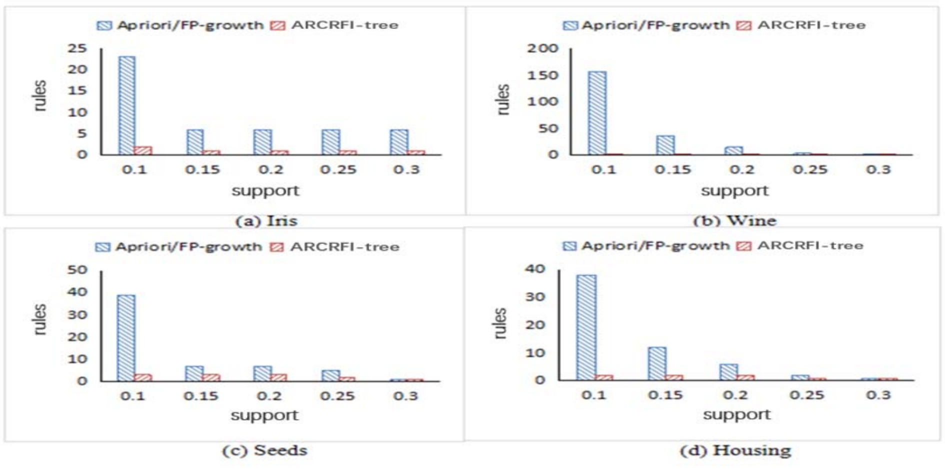

FIGURE II. THE NUMBER OF RULE

From the Figure 1, we can see that, (i)Under the different datasets, the proposed ARCRFI-tree algorithm needs the least running time. (ii)As the size of the support increases, the running time of all algorithms decrease, and the difference among them diminish, which are owing to the reduction of the frequent itemsets. (iii)At condition of the same support and different datasets, the running time of the ARCRFI-tree algorithm is always less than that of Apriori and FP-growth algorithm, which means that at different datasets the proposed ARCRFI-tree algorithm always keeps the optimal running time. From the Figure 1(a), with the growth of support, compared with the FP-growth and ARCRFI-tree, the running time of Apriori algorithm diminishes significantly, but is still greater than that of others. From the Figure 1(b), when the support is greater than 0.2, the running time of each algorithm is small. When the support is less than or equal to 0.2, the running time of the ARCRFI-tree algorithm is much less than that of the Apriori algorithm. From the Figure 1(c), the Apriori algorithm takes long time under the support of 0.1, in other cases, the running time of each algorithm is small. From the Figure 1(d), when the support is greater than 0.25, the difference in running time of each algorithm is little. In other cases, the running time of ARCRFI-tree algorithm is much less than that of the Apriori algorithm. Moreover, compared with the Apriori and FP-growth algorithm, the running time of ARCRFI-tree algorithm declines smoothly, which is because the generated frequent itemsets are reduced and the matrix operation is applied, so it is slightly affected by the changes of support and has good robustness.

From the Figure 2(a), when support is 0.1, the number of rules generated by Apriori/FP-growth algorithm is significantly greater than that of ARCRFI-tree algorithm. As the size of the support increases, the number of rules generated by Apriori/FP-growth algorithm is reduced because the frequent itemsets reduced, but still significantly greater than that of ARCRFI-tree algorithm. From the Figure 2(b), Figure 2(c) and Figure 2(d), with the growth of support, the number of rules generated by Apriori/FP-growth algorithm is reduced significantly, but still obviously greater than that of ARCRFI- tree algorithm. The number of rules generated by the ARCRFI-tree algorithm is less than that of the other algorithms, which is because that many redundancies are removed. In addition, the number of rules is insensitive to the changes of the support. All in all, based on the rules obtained by our algorithm, it is conductive to the applications of classification and regression prediction.

In order to verify the classification and regression prediction performance of the ARCRFI- tree, in this part, the rolling dataset comes from the real-world application was used. Moreover, 1000 samples were selected as training data and the other 500 samples were selected as test data. The proposed ARCRFI-tree was compared with BP [12], RBF [13], SVM [14], CART [15] and QART [10] in terms of training error and test error. The comparison results are shown in TABLE IV.

From the TABLE IV, the proposed ARCRFI-tree algorithm has a training error of 0.1750 and a test error of 0.1883, which outperforms other algorithms on accuracy, such as BP neural network, RBF neural network, SVM and CART algorithm. In addition, the training and test error of ARCRFI-TREE algorithm is relatively smaller than QART. All in all, the ARCRFI-tree algorithm has a high classification and regression prediction performance.

TABLE IV. THE COMPARISON OF ALGORITHMS ON ACCURACY

Algorithm	Training Error	Test Error
BP neural network	0.2064	0.4439
RBF neural network	0.2205	0.5753
SVM	0.9760	0.8263
CART	1.1040	1.0745
QATR	0.1858	0.1989
ARCRFI-TREE	0.1750	0.1883

V. CONCLUSION

In this paper, a novel association classification and regression algorithm based on frequent itemset tree is proposed. Firstly, the clustering algorithm is used to discretize the transaction data. Then, the discrete transaction database is transformed into Boolean matrix based on matrix operation. Taking the advantage of tree structure into account, a new frequent itemset tree structure is proposed, and all frequent itemsets can be obtained when the tree structure is constructed. Finally, the consequents of the rules are got based on the least square method, and then the classification and regression model is constructed to realize the classification and regression prediction for new sample data. To evaluate the performance and effectiveness of the proposed algorithm, we choose datasets of UCI benchmark databases and rolling dataset, and use a variety of algorithms to operate simulation experiments. The results demonstrate that the algorithm proposed in this paper has higher classification and regression prediction accuracy and mining efficiency.

ACKNOWLEDGMENT

This research work was supported by the National Natural Science Foundation of China (Grant No.61572073), National Key R&D Program of China (NO.2017YFB0306403) and the Fundamental Research Funds for the China Central Universities of USTB (No. FRF-BD-17-002A).

REFERENCES

[1] T Imielienskin, A Swami, R Agrawal. “Mining association rules between set of items in large databases,” Acm Sigmod Record, vol 22(2), pp. 207-216, 1993.

[2] Y E Shiqi, Z Sun, Z Zhao. “High-value Degree Association Rules Mining Algorithm Based on Boolean matrix,” Science & Technology Management Research, vol 34(6), pp. 188-191, 2014.

[3] H U Weihua, W Feng. “Improved Apriori algorithm based on decomposed transaction matrix,” Journal of Computer Applications, vol. (s2), pp.113-116, 2014.

[4] J Peng, X L Wang. An improved association rule algorithm based on Itemset Matrix and Cluster Matrix, in Proc. 7th International Conference on Computer Science & Education. IEEE, 2012, pp. 834-837.

[5] Z Zhou, J Wang. An Improved Matrix Sorting Index Association Rule Data Mining Algorithm, in Proceedings of the 33rd Chinese Control Conference. 2014, pp. 500-505.

[6] J Han, J Pei, Y Yin, R Mao. “Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach,” Data Mining & Knowledge Discovery, vol. 8(1), pp. 53-87, 2004.

[7] B Liu, W Hsu, Y Ma. “Integrating Classification and Association Rule Mining,” Proc of Kdd, vol. 1711, pp. 80--86, 1998.

[8] W Li, J Han, J Pei. CMAR: accurate and efficient classification based on multiple class-association rules, IEEE International Conference on Data Mining. IEEE, Vol 28, 2001, pp. 369-376.

[9] L Wang,W Lulu, F Dongmei. “A density-based fuzzy adaptive clustering algorithm,” Journal of University of Science and Technology Beijing, vol 36(11), pp. 1560-1565, 2014.

[10] W Ling , L Shulin, W Lulu. “Categorization and regression algorithm based on the quantitative association rule tree,” Chinese Journal of Engineering, vol 38(6), pp. 886-892, 2016.

[11] C. L. Blake and C.J.Merz, UCIR epository of machine learning databases, http://archive.ics.uci.edu/ml/datasets/ Glass + Identification. Irvine, CA: University of California, Department of Information and Computer Science 1998.

[12] R Xie,X Wang,Y Li, K Zhao. Research and application on improved BP neural network algorithm, in Proc. 5th Industrial Electronics and Applications. IEEE, 2010, pp. 1462-1466.

[13] G E Tsekouras, J Tsimikas. “On training RBF neural networks using input–output fuzzy clustering and particle swarm optimization,” Fuzzy Sets & Systems, vol 221(5), pp. 65-89, 2013.

[14] Y Zhao, J Sun. “A fast method to approximately train hard support vector regression,” Neural Networks the Official Journal of the International Neural Network Society, vol 23(10), pp. 1276-1285, 2010.

[15] L Rutkowski, M Jaworski, L Pietruczuk, P Duda. “The CART decision tree for mining data streams,” Information Sciences, vol 266(5), pp. 1-15, 2014.