Ant-Based Data Reduction Algorithm for Classification

[1]Prof. Svapnil Vakharia, [2]Mrs. Mansi Shah, [3]Kajal Mengar

[1,2]Assistant Professor, [3]M.E. Scholar
Department of Information Technology
[1]Gandhinagar Institute of Technology
[2,3]Kalol institute of technology

***Abstract*: Data reduction is the process of minimizing the amount of data that needs to be stored in a data storage environment. Data reduction can increase storage efficiency and reduce costs.It also decreases size of the training set presented to the algorithm by keeping only the most representative instances.**
In this paper, we introduce ADR-Miner with c4.5 and attribute selected classifier, novel data reduction algorithm that utilizes ant colony optimization (ACO). ADR-Miner is designed to perform instance selection to improve the predictive effectiveness of the constructed classification models. Empirical evaluations on 20 benchmark data sets with three well-known classification algorithms show that ADR-Miner improves the predictive quality of the produced classifiers.

***Keywords*: Ant Colony Optimization (ACO), Data Mining, Classification, Data Reduction, Instance Selection.**

1. Introduction

Data mining is the process of extracting insightful knowledge from large quantities of data either in an automated or a semi-automated fashion [1], [2]. Classification, a central problem in the field of data mining, is a type of supervised machine learning [2], [3]. Given a set of labelled instances of data (where the class of each instance is known), the aim of a classification algorithm is to capture the relationships between the input attributes and the corresponding class, and produce a model that is capable of predicting the class of new, unseen instances.

We analyzed several decision tree classification algorithms currently in use, including the ID3 [4] and C4.5 [2] algorithm as well as some of the improved algorithms [3] [5] [6] there after them. When these classification algorithms are used in the data processing, we can find that its efficiency is very low and it can cause excessive consumption of memory.

A. Decision Tree induction: Decision tree induction is the learning of decision trees from class labeled training tuples. A decision tree is a flowchart-like tree structure, where each internal node (non leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node. Suppose we want to buy the computer It represents the concept buys computer, that is, it predicts whether a customer likely to purchase a computer. Internal nodes are denoted by rectangles, and leaf nodes are denoted by ovals. Some decision tree algorithms produce only binary trees (where each internal node branches to exactly two other nodes), whereas others can produce non binary trees. C4.5 adopt greedy (i.e., non backtracking) approach in which decision trees are constructed in a top- down recursive divide and conquer manner. Most algorithms for decision tree induction also follow such top-down approach, which starts with a training set of tuples and their associated class labels. The training set is recursively partitioned into smaller subsets as the tree is being built.

2. Methodology

Steps of the System:
1. Selecting dataset as an input to the algorithm for processing.
2 .Selecting the classifiers
3. Calculate entropy, information gain, gain ratio of attributes. 4. Processing the given input dataset according to the defined algorithm of C4.5 data mining.
5. According to the defined algorithm of improved C4.5 data mining processing the given input dataset.
6. The data which should be inputted to the tree generation mechanism is given by the C4.5 and improved C4.5 processors. Tree generator generates the tree for C4.5 and improved C4.5 decision tree algorithm.

3. DATA MINING AND KNOWLEDGE DISCOVERY A.Attribute Selection Measure: The attribute selection measure provides a ranking for each attribute describing the given training tuples. The attribute having the best score for the measure is chosen as the splitting attribute for the given tuples. If the splitting attribute is continuous-valued or if we are restricted to binary trees then, respectively, either a split point or a splitting subset must also be determined as part of the splitting criterion. The tree node created for partition D is labeled with the splitting criterion, branches are grown for each outcome of the criterion, and the tuples are partitioned accordingly. There are two most popular attribute selection measures—information gain, gain ratio.

B. Classifiers: In order to mine the data, a well-known data mining tool WEKA was used. Since the data has numeric data type with only the classification as nominal leading to the category of labeled data set. Therefore it is needed to perform supervised data mining on the target data set. This narrowed down the choice of classifiers to only few, classifiers that can handle numeric data as well as give a classification (amongst a predefined set of classifications). Hence selecting C4.5 decision tree learning became obvious. The attribute evaluation was also performed in order to find out the gain ratio and ranking of each attribute in the decision tree learning. In case for some data set data mining could not produce any suitable result then finding the correlation coefficient was resorted to investigate if relation between attributes.

C. Entropy: It is minimum number of bits of information needed to encode the classification of arbitrary members of S.

D.Information Gain: It is simply the expected reduction in entropy caused by partitioning the examples according to the attribute .More precisely the information gain, Gain(S, A) of an attribute A, relative collection of examples S, is given by equation. Gain (A) = I (S1,S2, · · ·, Sm) − E (A) In other words gain (A) is the expected reduction in entropy caused by knowing the Value of attribute A. The algorithm computes the information gain of each attribute. With highest information gain is chosen as the test attribute for a given set.

E. Gain ratio: It differs from information gain, which measures the information with respect to classification that is acquired based on the same partitioning.

4. Ant colony Optimization (ACO)

Ant colony optimization (ACO) [7], is a search meta heuristic designed to tackle combinatorial optimization problems, inspired mainly by the foraging behaviour observed in ant colonies to find the shortest path between a food source and the nest. In essence, the search space of a given problem is first transformed into a graph of solution components, whereby a combination of these components would present a valid candidate solution to that problem. A population of ants then navigates this graph, selecting decision components to add to their path chosen throughout the graph in an attempt to build a candidate solution. In parallel to real life ants, the ants are trying to find the shortest path, where the " shortness" of their path corresponds to the " quality" of the constructed solution in the context of the problem at hand. The probability for an ant to select its next component is based on two properties: the heuristic advantage associated with it, and the amount of pheromone present. As the ants traverse the graph, they deposit pheromone on the nodes that they have chosen in proportion to the quality of the solution constructed by the ant. The more a decision component is chosen by ants in the colony (implying that it contributes well to the quality of solutions produced), the more pheromone is deposited on it; the higher the probability that it will be picked by ants in future iterations. The iterative process of building candidate solutions, evaluating their quality and updating pheromone values allows an ACO algorithm to converge on a near-optimal solutions.

5. Data Reduction

As mentioned earlier, data reduction is a vital pre-processing task for classification and its significance lies in that it removes noisy, outlier and other instances from the training data set that can be detrimental or misleading to the algorithm learning a model. In addition to improving accuracy, it also reduces the size of the training set before it is presented to the machine learning algorithm. The use of a reduced training set results in shorter training times for *eager-learning* classification algorithm, such as induction decision trees, classification rules and probabilistic models. For *lazy-learning* algorithms, such as nearest neighbour(s), the reduced data set decreases the time needed for arriving at the class of a new instance in question. In addition, a smaller data set would require less resources for storage and maintenance. One of the earliest algorithms for data reduction is Wilson editing, also known as editing nearest neighbour. This algorithm attempts reduction by going through the instances and removing those that are incorrectly classified by their nearest neighbours, typically considering the three closest neighbours. Two known extensions of Wilson editing are Repeated Edited Nearest Neighbour (RENN) and All *k*-Nearest Neighbours (All-*k*NN).

The IB2 and IB3 algorithms, part of the Instance-
Based learning (IB) family of algorithms, are incremental lazy learners that performed reduction by means of instance selection. With IB2, a new instance is added to the set of maintained instances by the lazy classifier if and only if it cannot be correctly classified by the set already maintained. At the end, this maintained set then becomes our reduced set. IB3 enforces a policy that removes instances from the maintained set if they contribute negatively to the classification. This is done by keeping track of how well instances in the maintained set classify instances in the training set. ICF achieves reduction over two phases: first it performs regular Wilson editing, with a neighbourhood size of 3 typically, then it builds two sets for each instance that still remains: a reachable set and a coverage set. The reachable set are those instances that include the current instance in their neighbourhoods. The coverage set are those instances that have the current instance as a neighbour. After having built those sets, the algorithm then removes those instances that have a reachable set larger than that of their coverage set. These instances are considered superfluous and their removal should not affect the quality of a classifier built using the remaining instances. The ICF algorithm has proven to be an effective data reduction algorithm in terms of maintaining the predictive power of the predictive models and reducing the size of the used training data.Therefore, we used it in our experiments for comparative evaluation with our introduced ACO-based data reduction algorithm, as described in the following section.

6. The ADR-Miner Overall Algorithm

The overall procedure for ADR-Miner can be seen in Algorithm.
We begin by initializing the pheromones on all the decision components to the value of 1 (line 4). This includes both the components for inclusion and exclusion for all instances in the data set to be reduced (i.e., the current trainingset). We then enter a *repeat−until* loop (lines 5 to 21) that is terminated when either of the following criteria are reached: we exhaust max_iterations number of iterations,or the colony has converged on a solution and no visible improvement has been observed over conv_iterations number

of iterations, where max_iterations and conv_iterations are external parameters.Within each iteration *t* of this outer loop, each *anta* in the colony constructs a candidate solution *Ra* (line 7), that is, a reduced set of instances. After a candidate solution is produced, a classification model *Ma* is constructed using the reduced set *Ra* and an input classification algorithm *g* (line 8). The quality of model *Ma* is evaluated (line 9), and if it is higher than that achieved by other ants in the current iteration *t*, it supplants an iteration best solution *Rtbest* (lines 10 to 13).
After all the ants in the colony complete the building of their solutions, the best ant in the iteration is allowed to update the pheromone trails based on *Rtbest*. This complies with the pheromone update strategy of the *MAX-MIN* Ant System [7], on which this algorithm is based. The iteration best solution *Rtbest* will supplant the best-so-far solution *Rbsf* if it is better in quality (lines 16 to 20). This process is repeated until the main loop exits, at which point, the best-sofar solution, *Rbsf* , observed over the course of the algorithm is returned as the output reduced set.

Algorithm 1 Pseudo-code of ADR-Miner.

```
1: begin
2: g ← classification_algorithm
3: I ← training_set;
4: InitializePheromones();
5: repeat
6: for a ← 1 to colony_size do
7: Ra ← anta.CreateSolution(I);
8: Ma ← ConstructModel(g,Ra);
9: Qa ← EvaluateModelQuality(Ma);
10: if Qa > Qtbest then
11: Qtbest ← Qa;
12: Rtbest ← Ra;
13: end if
14: end for
15: UpdateP heromones(Rtbest);
16: if Qtbest > Qbsf then
17: Qbsf ← Qbest;
18: Rbsf ← Rtbest;
19: t ← t + 1;
20: end if
21: until Convergence(conv_iterations) or t = max_iterations
22: return Rbsf ;
23: end
```

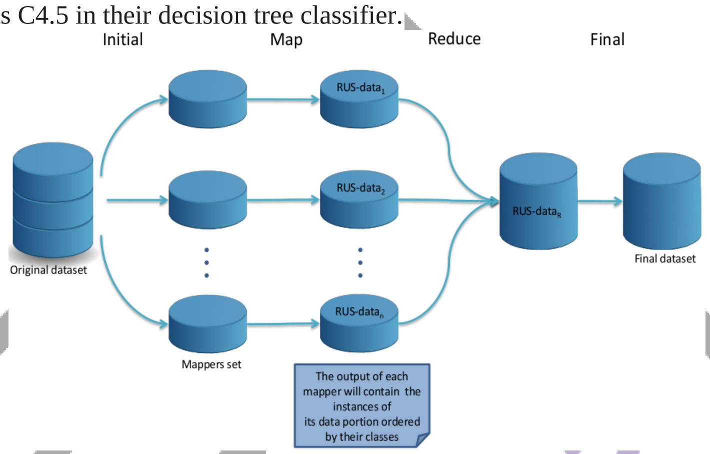
Algorithm 2 Soution Creation Procedure

```
1: begin
2: Ta ← φ; \* ant trail */
3: Ra ← φ; \* reduced data set */
4: for i ← 1 to |I| do
5: dvi ← SelectDecisionComponent();
6: Ta ← Ta ∪ dvi
7: if dxi = dTrue i then
8: Ra ← Ra ∪ Ii
9: end if
10: end for
11: return Ra
12: end
```

7. C4.5 ALGORITHM

C4.5 is an algorithm used to generate a decision tree developed by Ross qiunlan. Many scholars made kinds of improvements on the decision tree algorithm. But the problem is that these decision tree algorithms need multiple scanning and sorting of data collection several times in the construction process of the decision tree. The processing speed reduced greatly in the case that the data set is so large that cannot fit in the memory .At present, the literature about the improvement on the efficiency of decision tree classification algorithm For example, Wei Zhao, Jamming Su in the literature [7] proposed improvements to the ID3 algorithm, which is simplify the information gain in the use of Taylor's formula. But this improvement is more suitable for a small amount of data, so it's not particularly effective in large data sets. Due to dealing with large amount of datasets, a variety of decision tree classification algorithm has been considered. The advantages of C4.5 algorithm is significantly, so it can be choose. But its efficiency must be improved to meet the dramatic increase in the demand for large amount of data. A. Pseudo Code : 1. Check for base cases. 2. For each attribute a calculate: i. Normalized information gain from splitting on attribute a. 3. Select the best a, attribute that has highest information gain. 4. Create a decision node that splits on best of a, as rot node. 5. Recurs on the sub lists obtained by splitting on best of a and add those nodes as children node. B. Improvements from ID3 algorithm: C4.5 made a number of improvements to ID3. Some of these are follows: 1.Handling both continuous and discrete attributes - In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it. 2. Handling training data with missing attribute values - C4.5 allows attribute values to be marked as ? for missing. Missing attribute values are simply not used in gain and entropy calculations. 3. Handling attributes with differing costs. 4. Pruning trees after creation - C4.5 goes back through the tree once it's been created and attempts to remove branches that do not help by replacing them with leaf nodes

Why use C4.5?
Arguably, the best selling point of decision trees is their ease of interpretation and explanation. They are also quite fast, quite popular and the output is human readable.
Where is it used?
A popular open-source Java implementation can be found over at OpenTox. Orange, an open-source data visualization and analysis tool for data mining, implements C4.5 in their decision tree classifier.



Working of c4.5

8. Literature Review

Sr No	Method Name	Description	Advantage
1	IB2,IB3	Incremental lazy learners that performed reduction by means of instance selection.	IB3 enforces a policy that removes instances from the maintained set if they contribute negatively to the classification.
2	DROP	Performs reduction by considering whether removing instance would result in misclassification of its neighbours.	The algorithm does this iteratively for each instance in the data set, and the remaining set of instances after this whittling then becomes the reduced data set.
3	C4.5	C4.5 adopt greedy (i.e., non backtracking) approach in which decision trees are constructed in a top-down recursive divide and conquer manner	Handling both continuous and discrete attributes. Handling training data with missing attribute values. Handling attributes with differing costs.Pruning trees after creation

9. Conclusions

In this paper we introduce ADR miner algorithm with three classification algorithm. Results were achieved, when paired with C4.5 classifiers. The results also indicate that the ADR-Miner algorithm establishes a statistically significant edge over random and greedy searches when paired with Attribute Selected Classifier models used. Furthermore, attribute reduction alongside instance reduction can be integrated into the ADR-Miner algorithm. One can also explore performing the reduction using one classification algorithm and then building the final model utilizing another classification algorithm.

10. Acknowledgment

I am extremely obliged to my guide Mansi Parmar and Mr. Lokesh Gagnani devoid of them guidance the work would not have happened and They supported me to solve my difficulties arise and give Valuable suggestions. I would like to pay my sincere gratitude for them endless motivation and support in progess and success of this survey work.

References

[1] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, 2nd ed. Addison Wesley, 2005. *Tools and Techniques*. San Francisco, CA, USA: Morgan Kaufmann, 2010.
[2] S. F. Chen, Z. Q. Chen, Artificial intelligence in knowledge engineering [M]. Nanjing: Nanjing University Press, 1997.
[3] Z. Z. Shi, Senior Artificial Intelligence [M]. Beijing: Science Press, 1998.
[4] D. Jiang, Information Theory and Coding [M]: Science and Technology of China University Press, 2001.
[5] M. Zhu, Data Mining [M]. Hefei: China University of Science and Technology Press, 2002.67-72.
[6] A. P. Engelbrecht., A new pruning heuristic based on variance analysis of sensitivity information [J]. IEEE Trans on Neural Networks, 2001, 12(6): 1386-1399
[7]T. Stützle and H. Hoos, " MAX-MIN Ant System and local search for thetraveling salesman problem,
" *Evolutionary Computation, 1997., IEEE International Conference on*, pp. 309– 314, 1997.