Ultra Scientist Vol. 27(3)B, 203-208 (2015).

Predictive Model for Water Quality data with Weka tool

ANIL RAJPUT, PURSHOTAM SHARMA and NEERAJ BHARGAVA

[1]CSA Govt. PG Nodal College, Sehore, MP (India)
[2]Vidhya Nagar, Bhopal, MP (India)
[3]Dept. of Computer Appl. MDS University, Ajmer Raj. (India)

(Acceptance Date 20th December, 2015)

Abstract

In this paper we present the experimental work with WEKA tool. We have use decision tree techniques for generating a predictive model with water quality data.

1. Introduction

In general, classification involves examining the features of new objects and trying to assign it to one of the predefined set of classes. Given a collection of records in a data set, each record consists of a group of attributes; one of the attributes is the class. The goal of classification is to build a model from classified objects in order to classify previously unseen objects as accurately as possible.

2. Methods

2.1 C4.5

This algorithm is a successor to ID3 developed by Quinlan Ross[11]. It is also based on Hunt's algorithm.C4.5 handles both categorical and continuous attributes to build a decision tree. In order to handle continuous attributes, C4.5 splits the attribute values into two partitions based on the selected threshold such that all the values above the threshold as one child and the remaining as another child. It also handles missing attribute values. C4.5 uses Gain Ratio as an attribute selection measure to build a decision tree. It removes the biasness of information gain when there are many outcome values of an attribute.

2.2 The J48 :

Decision tree classifier follows the following simple algorithm. In order to classify a new item, it first needs to create a decision tree based on the attribute values of the available training data. So, whenever it encounters a set of items (training set) it identifies the attribute that discriminates the various instances most clearly.

2.3 Decision Tree :

Decision tree-based methods are popular methods for use in a data mining

context. The decision tree classifier uses a hierarchical or layered approach to classification. Each vertex in the tree represents a single test or decision. The outgoing edges of a vertex correspond to all possible outcomes of the test at that vertex. These outcomes partition the set of data into several subsets, which are identified by every leaf in the tree. A leaf of the tree specifies the expected value of the categorical attribute for the records described by the path from the root to that leaf. Learned trees can also be represented as sets of if then-else rules.[3]

Most algorithms that have been developed for decision trees are based on a core algorithm that uses a top-down, recursive, greedy search on the space of all possible decision trees. This approach is implemented by ID3 algorithm2 [8,9] and its successor C4.5 [5-7]. C4.5 is an extension of ID3 that accounts for unavailable values, continuous attribute value ranges, pruning of decision trees, and rule derivation. The rest of this section discusses some important issues in decision trees classifiers.

2.4 Cross-Validation :

Cross-validation is a technique to eliminate the occurrence of overfitting. The main idea of cross-validation is to estimate how well the current hypothesis will predict unseen data. This is done by randomly dividing the data into two subsets, training and test. Usually, the test subset is a fraction of all of the data, *i.e.*, 10%.

2.5 Pruning :

The principal alternative of stop-splitting is pruning (Duda *et al.*, 2001). One approach, called *reduced-error pruning*[10], sets each node in the decision tree to be candidate for pruning. "Pruning a decision node consists of removing the subtree rooted at that node, making it a leaf node, and assigning it the most common classification of the training examples affiliated with that node. Nodes are removed only if the resulting pruned tree performs no worse than the original over the validation set."[3]. In C4.5, [6] applied a successful technique for finding high accuracy hypotheses during the pruning process, which is called *rule post pruning*.

2.6 Evolutions of model :

A crucial issue in machine learning is performance evaluation. This means defining one or more synthetic measures which summarize the behavior of the model — in other words, how much the model corresponds to the data. In a classification task, the most immediate performance measure is accuracy, which is the fraction of instances correctly classified. But this is far from giving us all the information we could need.

2.7 Confusion Matrix :

Consider a binary classification problem (with only two classes: positive and negative) on a dataset of N examples. In this case, a confusion matrix is used as a basis for performance evaluation.

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

Table 2.7.1 view of Confusion matrix

2.8 Algorithmic Framework for Decision Trees :

Induction of an optimal decision tree from a given data is considered to be a hard task. It has been shown that finding a minimal decision tree consistent with the training set is NP–hard[1]. Moreover, it has been shown that constructing a minimal binary tree with respect to the expected number of tests required for classifying an unseen instance is NP–complete[2]. Even finding the minimal equivalent decision tree for a given decision tree[12] or building the optimal decision tree from decision tables is known to be NP–hard[4].

2.9 Experimental Process and Setup :

Since the data mining software used to generate association rules accepts data only in arff format, the researcher first converted the data on Ms Excel file into comma separated text format and then to arff format. Data in arff format is then given to Weka software **Convert a TEXT file containing data into ARFF** format (readable by Weka tool):

1- Open Excel.
2- Open the text file from the menu, a window will open that asks you "what is the delim for the data in this file" depending on how the data is separated, choose tab, comma or space as delim. This will produce the columns from your data.
3- Add a new row at the top of the Excel worksheet.
4- Enter the header for each column (e.g. Class, attribute 1, etc..)
5- Save the file as CVS format
6- Open Weka tool
7- Open the CVS file that you created in step 5
8- Save the file as ARFF using the save button
9- Open the new ARFF file using a simple text editor
10- Check all the attribute types. Make sure nominal values are not confused with numeric values
11- After you made all the changes save the file as ARFF

Experimental Setup

=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: Book1-weka.filters. unsupervised. attribute. Remove-R1-weka.filters.unsupervised.attribute.Remove-R2 weka.filters.supervised.attribute. Discretize-V-Rfirst-last-weka.filters.supervised.attribute.Discretize-V-Rfirst-last weka.filters.supervised.attribute. Discretize-V-Rfirst-last

Instances: 30
Attributes: 17
St_name
Total_Alkalinity
Carbonat_ alkalinity
Bi_Carbo_natealkalinity
Total_Hardness
Ca_hardness
Mg_hardness
Calcium_content
Magnesium_content
Chloride
Phosphate
Total_Phosphorus
Org_Phosphorus
Nitrate
BOD
COD
Period
Test mode: evaluate on training data

=== Classifier model (full training set) ===
J48 pruned tree

Phosphate <= 0.503
| Nitrate <= 1.227
| | Org_Phosphorus <= 0.496
| | | Phosphate <= 0.176
| | | | Nitrate <= 0.324: C (3.0)
| | | | Nitrate > 0.324: B (4.0/1.0)
| | | Phosphate > 0.176: A (5.0/1.0)
| | Org_Phosphorus > 0.496: C (6.0)
| Nitrate > 1.227: B (6.0)
Phosphate > 0.503: A (6.0)
Number of Leaves : 6
Size of the tree : 11
Time taken to build model: 0.03 seconds
=== Evaluation on training set ===
Time taken to test model on training data: 0 seconds
=== Summary ===
Correctly Classified Instances 28 93.3333 %
Incorrectly Classified Instances 2 6.6667 %
Kappa statistic 0.9
Mean absolute error 0.0689
Root mean squared error 0.1856
Relative absolute error 15.5 %
Root relative squared error 39.37 %
Coverage of cases (0.95 level) 100 %
Mean rel. region size (0.95 level) 43.3333 %
Total Number of Instances 30
=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.050	0.909	1.000	0.952	0.929	0.990	0.964	A
	0.900	0.050	0.900	0.900	0.900	0.850	0.978	0.937	B
	0.900	0.000	1.000	0.900	0.947	0.926	0.993	0.977	C
Weighted Avg.	0.933	0.033	0.936	0.933	0.933	0.902	0.987	0.959	

=== Confusion Matrix ===

Predicted Class				
	A	10	0	0
	B	1	9	0
	C	0	1	9
		A	B	C

Table Confusion matrix 2.9.1

Table 2.9.1 is the confusion matrix for classified sample data, representing a measure of accuracy for each class. While the training data for three classes show user accuracy of 100% and 90.0% and 90% respectively.

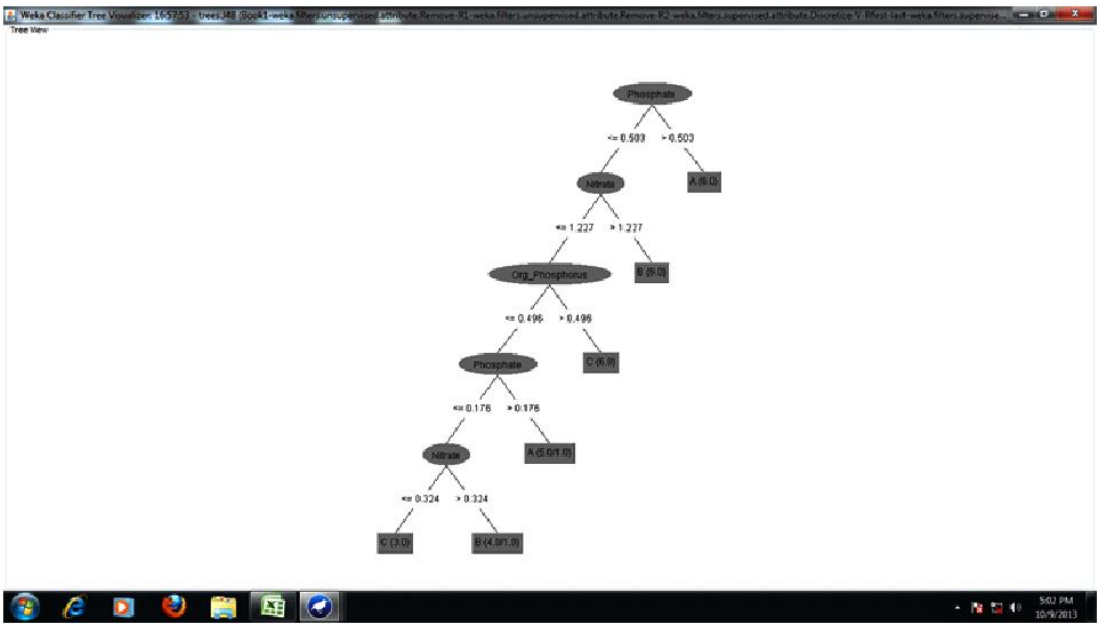

Table 2.9.2 Explanation of WEKA J48 parameters.

Parameters	Description	Status
Binary Splits	Whether to use binary splits at each node to built	Used
Confidence factor	The confidence factor used for prunling smallerconfidence factor implies more pruning, the default value is 0.25	0.25
Number of objects	Mininmum number of instances per leaf, if less number of samples are present in one leaf than the assigned value, the leaf will not be considered as a class.	5
Number of folds	Determines the amount of data used for reduced error pruning, one fold is used for pruning, rest is used for growing the tree.	Value set to 6, I fold is used for pruning and 5 fold is used for growing the tree
Size of tree	Number of nodes	11
Number of leaves	This is same as number of rules	06

3.0 Conclusion

The classification accuracy is the most popular performance evaluation measure used in predictive knowledge discovery where the goal of learning is prediction or classification. The classification accuracy measures the proportion of correctly classified cases. In binary classification problems using the confusion matrix notation, the accuracy is computed.

3.1 References

1. Hancock T. R., Jiang T., Li M., Tromp J., Lower Bounds on Learning Decision Lists and Trees. Information and Computation *126(2)*, 114-122 (1996).
2. Hyafil L. and Rivest R.L., Constructing optimal binary decision trees is NP-complete. Information Processing Letters, *5*(1),15-17 (1976).
3. Mitchell, T. "Decision Tree Learning", in T. Mitchell, Machine Learning, The McGraw-Hill Companies, Inc., pp. 52-78 (1997).
4. Naumov G.E., NP-completeness of problems of construction of optimal decision trees. Soviet Physics: Doklady, *36(4)*, 270-271 (1991).
5. Quinlan, J. R. and Rivest, R. L., Inferring Decision Trees Using The Minimum Description Length Principle. Information and Computation, *80*, 227-248 (1989).
6. Quinlan, J.R., C4.5: Programs for Machine Learning, Morgan Kaufmann, Los Altos, (1993).
7. Quinlan, J. R., Unknown attribute values in induction. In Segre, A. (Ed.), Proceedings of the Sixth International Machine Learning Workshop Cornell, New York. Morgan Kaufmann (1989).
8. Quinlan, J.R., Decision Trees and Multivalued Attributes, J. Richards, ed., Machine Intelligence, V. 11, Oxford, England, Oxford Univ. Press, pp. 305-318, (1988).
9. Quinlan, J.R., Induction of decision trees, Machine Learning *1*, 81-106 (1986).
10. Quinlan, J.R., Simplifying decision trees, International Journal of Man Machine Studies, *27*, 221-234 (1987).
11. Ross Quinlan originally developed ID3 at the University of Sydney. He first presented ID3 in 1975 in a book, Machine Learning, Vol. 1, no. 1. ID3 is based off the Concept Learning System (CLS) algorithm.
12. Zantema, H. and Bodlaender H.L., Finding Small Equivalent Decision Trees is Hard, International Journal of Foundations of Computer Science, *11*(2), 343-354 (2000).