ASSIGMENT 2

Asam Mahmood

Florida Atlantic University

PROF. TAGHI M KOSHGOFTAR

Course CAP 6673 Data Mining and Machine Learning

Table of Contents

Figures

1.

Introduction

Classification is a supervised learning concept in machine learning[1]. It breaks up instances specified in sets of data down to groups or classes. In our class assignment in CAP6673 we have a class of (fp,nfp) that was predicted using a decision tree. The decision tree is using the J48 algorithm which was specified by requirements in assignment and we are using a machine learning tool named weka as a tool in our analysis and to develop the model. J48 algorithm in WEKA is implementation C4.5[1]. The algorithm considers all possibilities and validates then splits the data set in its implementation and then chooses the best test that gives it them most information gain from[1].C4.5 algorithm is used by many in the field of machine learning[1].

Input

Our data FIT and Test data sets were given to us as a requirement to use. There are 9 attributes in and 188 instances in FIT data set. The Test data set has 94 instances and 9 attributes. Class attribute in each set has a requirement when attribute is less than 2 faults are considered nfp, and modules with 2 or more faults are considered fp .It was used in our previous assignment where we engineered the input.

Table A – number of leaves and nodes highlighted yellow has the highest number

	Confidence factor	*LEAF*	*NODE*
Pruned	.25	8	15
Un-Pruned	.25	11	21
Pruned	.01	2	3

1.1 Part 1

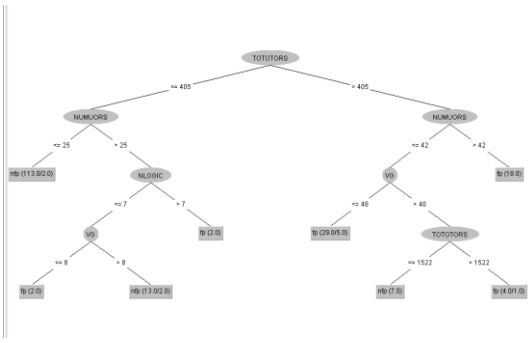
Above in table A shows you the number of leaves and nodes in the decision tree with 10-fold cross validation set on the fit- classification dataset and test data set output. The unpruned tree has the largest number of leaves and node which is highlighted in yellow in table above. Un-Pruned trees are larger because the tree is created according to the algorithm and when pruning is set to true an extra step is taken to look at nodes/branches that can be removed without affecting accuracy and performance. Which I will highlight more in part 2 below.

Below Decision Tree 1a displays the decision tree of the initial model with confidence factor of .25 and Decision Tree 2a shows the decision tree with the setting of unpruned to false. Each Circle is a node with attribute/feature to be validated against. Decision trees work from a top-down approach. The very first attribute which is TOTOTORS is the root node which will be the first value validated against. If the value if <=405 it goes down to the left node NUMUORS. If value > 405 it traverses to the right. It then continues down and traverses the tree and validates against nodes which steer the path until it reaches the rectangular box which is called a leaf.

The leaf has two numbers within it. The first value in the leaf is the number value of the 188 instances that meet the requirement leading up to that box. The second number represents the misclassified instances that lead up to the box [1] page 376. I realized in testing and iterating that when the second value is missing that it means the leaf has classified all instances correctly. I also validated this theory when reading more into it in the text which can be found in page 196 [1]. We also used Weka J48 algorithm to derive our tree which is the C4.5 decision tree learner[1] page 404.

Currently in Table B below I managed to calculate the TYPE I error and TYPE II error using the confusion matrix. Type I error when setting prune to false and confidence factor was set .25 was .104 and the type II error is .245. for Test you can see I added under test column TYPE I is .134 and TYPE II came out to be .370. The reason Type I and TYPE II errors are important because it tells us our false positive and false negatives. It depends on use case to which one is better or worse. You can see below in Table B calculations from confusion matrix and validated in weka. You see can see correctly classified metrics as well. I also included the ROC curve value which are very useful for exploring tradeoffs among different classifiers over a range of costs [1] page 173. ROC value is not ideal evaluating machine learning models in situations with known error costs. Also, people use the AUC because the larger the area the better the model which the author speaks about as well on page 173. The Un-Pruned and Pruned had almost identical metrics but un-pruned AUC was slightly larger giving it the edge.

Decision Tree 1a – Decision Tree of Pruned



Decision Tree 2a – Decision Tree of Un-Pruned Model

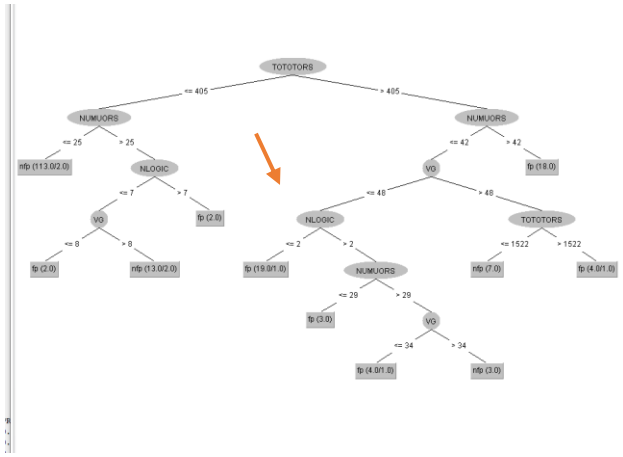


Table B – This table shows misclassification, Confusion Matrix, Correctly Classified metrics, Incorrectly Classified Metrics, Type I error, and Type II error and show in rate %. Also displayed it the ROC value which is the area under roc curve. Also displayed Is the RMSE which is Root mean squared error. I also displayed the MAE which is mean absolute error

	Pruned=false				***Unpruned=true***				***Pruned=false***			
	C = .25				***C = .25***				***C = .01***			
	fit		***test***		***Fit***		***test***		***fit***		***test***	
Confusion Matrix	40 13 14 121		17 10 9 58		40 13 14 121		17 10 9 58		42 11 12 123		23 4 10 57	
Misclassification	14.3617		20.2128		14.3617		20.2128		12.234		14.8936	
Correctly Classified	161		75		161		75		165		80	
Correctly Classified %	85.6383		79.7872		85.6383		79.7872		87.766		85.1064	
In-Correctly Classified	27		19		27		19		23		14	
In-Correctly Classified %	14.3617		20.2128		14.3617		20.2128		12.234		14.8936	
Type I error	14/135	0.104	9/67	.134	14/135	0.104	9/67	.134	12/135	0.089	10/67	0.149
Type II Error	13/53	0.245	10/27	0.370	13/53	0.245	10/27	.370	11/53	0.208	4/27	0.148
MAE	0.1665		0.2153		0.1626		0.2178		0.1772		0.2146	
RMSE	0.354		0.4182		0.3526		0.434		0.3282		0.3407	
ROC	0.823		0.686		0.831		0.686		0.838		0.851	

1.2 Part 2

The unpruned tree has the node includes “NLOGIC” node when VG <= 48. I highlighted with an arrow pointing at it as reference *in Decision Tree 2a*.In the pruned model when VG <=48 fp can be (29.0/5.0). The pruned model simplified it heavily. This can also reduce the complexity of the tree as a whole and makes it more inaccurate [1] PAGE 195. Looking at the metrics of un-pruned tree and pruned tree it look the same except the unpruned tree has 3 more nodes and 4 leaves more than pruned. NLOGIC NUMUORS and VG are the nodes. Which determines the part that was pruned.

This reflects what the textbook outlines when speaking about subtree raising and subtree replacement and differences. [1*] The idea is to select some subtrees and replace them with single leaves*. The whole sub tree is replaced with a single leaf which shows on our tree above this is a ***subtree replacement example***. This post pruning process makes this decision rationally by estimating the error rate at an internal node or leaf .[1] Citing the *author If we had such an estimate, it would be clear whether to replace, or raise, a particular subtree simply by*

comparing the estimated error of the subtree with that of its proposed replacement. Comparing the error estimate will determine this. This explains why and how it was pruned and where in the process it is pruned. I will go further in the confidence factor how estimate error rate is determined.

1.3 Confidence Factor

In table B you can see the values of initial model with confidence factor set to .25. The newer model at the end of table B has it set to .01. The very first table(table A) you can see all leaves and nodes values. I came to conclusion in my testing and validation that when you increase confidence factor the higher the complexity of the tree.

In *Decision Tree 1a diagram* above you can see the decision tree when its initially at .25 and beneath in *Decision Tree 3a diagram* you can see decision tree when the cf (confidence factor) is set at .01. The conclusion I made was in my testing and investigation/research dropping the confidence factor to a lower value leads to more of the pruning of the tree(This is my observation in testing). Which in turn leads to less nodes and leaves. It feels less accurate you can see below it ended up pruning the nodes "NUMOURS" and subsequential nodes down and coming to conclusion that when TOTOTOS <= 405 nfp(130.0/8.0) and TOTOTORS > 405 fp(58.0/13.0).

The TYPE I error can be seen in *table b* .089 and type II error .208 with test evaluation TYPE I error being .149 and TYPE II error being .148. The main difference you can see between part 1 and and the new tree is that it replaces both subtrees with two leaves. It did this because of the error estimate values in the post pruning process. Which is the same math as before. The confidence factor in Weka is the effectiveness of post pruning process. This generated tree also reflects a drastic pruning example and the fact of how lower confidence factor mean more pruning [1] which is described in textbook in page 196 and page 199. The justification of what it prunes come from C4.5(J48) classifier using the estimate error to decide what to prune and not in post pruning process.

The formula below is described in the textbook or estimate error or upper confidence limit. E is the number of errors out of the total instance N, F = (3/N), and z the related z-score from the standard normal distribution[1] page 195. When the confidence factor gets smaller the Z value gets larger. When that happens error calculation at *e* gets smaller because the denominator itself is getting larger at a quicker rate than the numerator. Error is compared to the parent node and the sub - tree node is removed when parent is smaller. In [1] page 199 explains how this results in more drastic pruning.

Error Estimate Formula(Upper Confidence Limit) *– [1] Page 195*

$$e = \frac{f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}.$$

Note the use of the + sign before the square root

Decision Tree 3a *– Un-Pruned tree when confidence factor is set .01*

Table C – cost sensitivity

Cost Matrix	Confusion Matrix	Type I	Type II	Correctly Classified instances	Correctly Classified %	In-Correctly Classified instances	In-Correctly Classified %	Mean Absolute Error	ROC	Relative Absolute Error
0 .5 1 0	37 16 9 126	.067	.302	163	86.7021	25	13.2979	0.163	0.876	40.139%
0 .75 1 0	46 7 14 121	0.104	0.132	167	88.8298	21	11.1702	0.1445	0.904	35.5851%
0 1 1 0	40 13 14 121	0.104	0.245	161	85.6383	27	14.3617	0.1665	0.823	40.9949
0 1.5 1 0	45 8 15 120	0.111	0.151	165	87.766	23	12.234	0.1407	0.893	34.6502
0 2 1 0	46 7 20 115	0.148	0.132	161	85.6383	27	14.3617	0.1529	0.890	37.6558
0 2.5 1 0	45 8 21 114	0.156	0.151	159	84.5745	29	15.4255	0.1709	0.849	42.075
0 3 1 0	46 7 25 110	0.185	0.132	156	82.9787	32	17.0213	0.1908	0.842	46.9913
0 4 1 0	46 6 23 112	0.170	0.113	159	84.5745	29	15.4255	0.1796	0.843	44.2309
0 5 1 0	46 6 31 104	0.230	0.113	151	80.3191	37	19.6809	0.2214	0.808	54.5074
0 6 1 0	46 7 32 103	0.237	0.132	149	79.2553	39	20.7447	0.2264	0.816	55.755

Table D – cost sensitivity test evaluation

Cost Matrix	Confusion Matrix	Type I	Type II	Correctly Classified instances	Correctly Classified %	In-Correctly Classified instances	In-Correctly Classified %	Mean Absolute Error	ROC	Relative Absolute Error
0 .5 1 0	19 8 4 63	0.060	.296	82	87.234	12	12.766	0.1588	0.874	38.9047
0 .75 1 0	19 8 4 63	0.060	0.296	82	87.234	12	12.766	0.1468	0.870	35.9646
0 1 1 0	17 10 9 58	0.134	0.370	75	79.7872	19	20.2128	0.2153	0.686	52.736
0 1.5 1 0	45 8 15 120	0.164	0.296	75	79.7872	19	20.2128	0.2074	0.852	50.8131
0 2 1 0	20 7 7 60	0.104	0.259	80	85.1064	14	14.8936	0.1602	0.777	39.2573
0 2.5 1 0	20 7 7 60	0.104	0.259	80	85.1064	14	14.8936	0.1619	0.777	39.6715
0 3 1 0	22 5 15 52	0.224	0.185	74	78.7234	20	21.2766	0.2204	0.809	53.9922
0 4 1 0	21 6 13 54	0.194	0.222	75	79.7872	19	20.2128	0.2154	0.794	52.7655
0 5 1 0	24 3 15 52	0.224	0.111	76	80.8511	18	19.1489	0.2168	0.851	53.1248
0 6 1 0	24 3 15 52	0.224	0.111	76	80.8511	18	19.1489	0.2205	0.851	54.0195

1.4 Cost Sensitivity

You can see above in table type I increases as Type II decreases. As the requirement from instruction states TYPE II error is more serious than a type I error in our use case condition. So, the best would be lowest TYPE II with a balanced TYPE I which by diffing them can help you find the smallest difference as well. You can also see on graphs below the optimal most balance one is between the ratio and 2/1 and 4/0. You can see from the graph and tables above as I increase cost it greatly increases TYPE I error and lowers TYPE II. The best result and most balance result are when I set the cost to 2. You can also see in graph below it reflects that where it is about to intersect and where it is not. I also placed Test evaluation graph as well to below with the best balance lowest difference lowest type II.

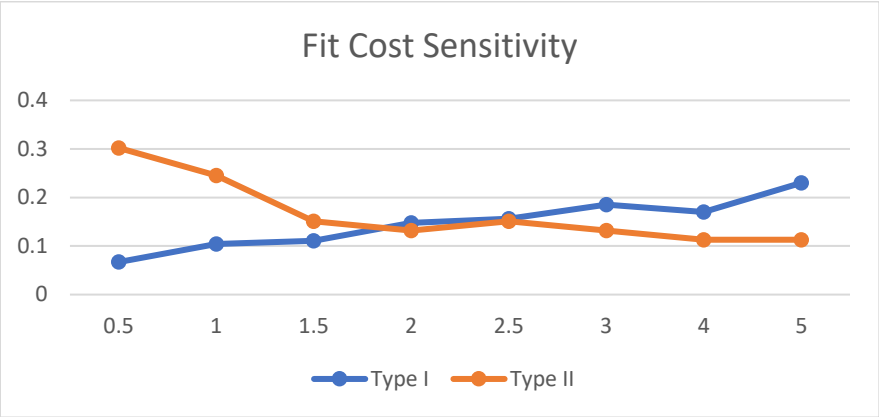
FIT

Cost Matrix	Confusion Matrix	Type I	Type II	Correctly Classified instances	Correctly Classified %	In-Correctly Classified instances	In-Correctly Classified %	Mean Absolute Error	ROC	Relative Absolute Error
0 2 1 0	46 7 20 115	0.148	0.132	161	85.6383	27	14.3617	0.1529	0.890	37.6558

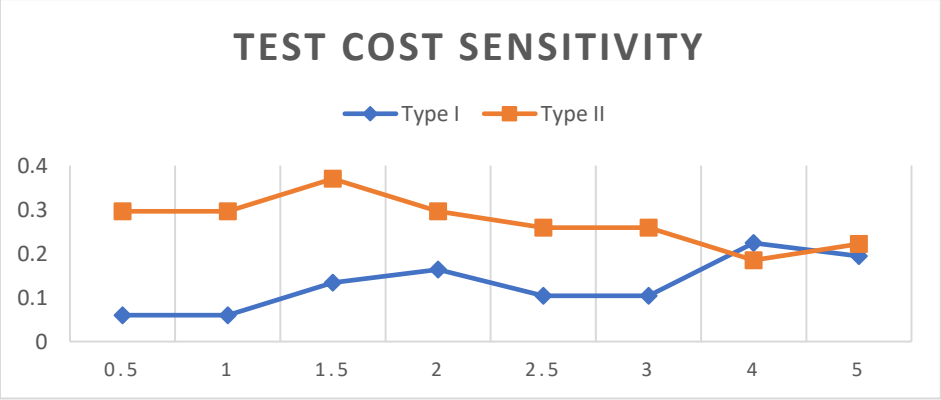
Test

Cost Matrix	Confusion Matrix	Type I	Type II	Correctly Classified instances	Correctly Classified %	In-Correctly Classified instances	In-Correctly Classified %	Mean Absolute Error	ROC	Relative Absolute Error
0 4 1 0	21 6 13 54	0.194	0.222	75	79.7872	19	20.2128	0.2154	0.794	52.7655

Graph 1 - FIT



Graph 2 -Test



1.5 Results

In Table B When comparing results between the models when confidence factor is set to .01 it has better TYPE I at .089 and TYPE II error at .208 then first two initial models which meets the requirements and has a valid type I errors. But When Iterating through values in cost sensitivity validation in part 4 you can see that the optimal when set to 2 in FIT cost sensitivity table C TYPE I at .148 and TYPE II at .132 it is more significantly better than the other three models. So, we can safely assume the best model is the most optimal one from part 4 FIT. This also proves when iterating through value in the cost matrix it helps achieve a more improved classification and helps lowering the Type II error which is the harmful one in our use case.

References

[1] Witten, Ian H. et al. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2017