INFSCI 2725 Data Analytics

Assignment 7 – Logic-based approaches

Date: 11/24/2015
Student name: Ting LI, Hongyuan Cui
pittUserLogin: til42, hoc27
Tutor name: Marek Druzdzel

Executive Summary

This report applies logic-based approaches on Iris dataset and Congressional Voting Data. 10-fold cross validation is used to test the classification models. For Iris Data, 18 classification models are trained and it seems that accuracy given by those logic-based approaches do not vary a lot. Their classification accuracy are basically around 94%-96%. LMT that builds classification trees with logistic regression functions at the leaves renders the highest classification of 97.33%. For congressional voting data, 15 classification models are conducted to train the dataset and accuracy given by those logic-based approaches also do not vary a lot. Their classification accuracy are roughly around 92%-97%. FT Tree that builds classification trees with minNumInstances of 150 renders the highest classification of 97.01%.

Dataset: Iris Data

Iris dataset has 150 rows where there is no missing value in iris dataset and all attributes are numeric. Table 1 illustrates the performance for different classification approaches. Row 1 is about six classification approaches under rules including Decision Table, JRip, NNge, OneR, PART and Ridor. The rest 12 is approaches under trees. The cells marked with gray are approaches that can not be applied on numerical attributes, thus discretizing is used. OneR is a class using the minimum-error attribute for prediction. Its accuracy remains at 94%, changing the number of bucket size did not make a difference. ID3 is a class for constructing an unpruned decision tree based on the ID3 algorithm, and its accuracy is 94% and there is no other parameter on which we can manipulate. We can also observe that OneR and ID3 are the two methods in the above table

that give us lowest accuracy, which would possibly ascribe to the discretizing process.

Table 1.Performance Summary for Iris Dataset

Approach (rules)	Decision Table	JRip	NNge	OneR	PART	Ridor
Accuracy	96%	96%	96%	94%	96%	95.3%
Approach (trees)	BF Tree	FT Tree	ID3	J48	J48graft	LADTree
Accuracy	96.67%	96.67%	94%	96%	95.3%	95.3%
Approach (trees)	LMT	NBTree	Random Forest	Random Tree	REP Tree	SimpleCart
Accuracy	97.33%	94.67%	96%	96.67%	95.3%	95.3%

Although for the rest approaches discretization is not necessary, the cells marked with green are those I found discretizing the attributes render better outcome. Decision Table is a class for building and using a simple decision table majority classifier, and genetic Search gives the highest accuracy. JRip implements a propositional rule learner. NNge is a nearest-neighbor-like algorithm using non-nested generalized exemplars, and its accuracy remains unchanged after modifying its parameters. Random Forest is a class for constructing a forest of random trees. While Random Tree constructs a tree considering K randomly chosen attributes at each node. 5-fold back-fitting gives the best accuracy of 96.67%.

PART uses separate-and-conquer for generating a PART decision list. It builds a partial C4.5 decision tree in each iteration and makes the "best" leaf into a rule. By increasing the number of objects from 3 to 5, correctly classified instance increases from 94% to 95.33%. If at the same time pruning the tree, the accuracy reaches 96%. However further increasing this number of objects resulted in a decreasing accuracy. FT Tree is a classifier for building 'Functional trees', which are classification trees that could have logistic regression functions at the inner nodes and/or leaves. The accuracy increases from 94.6% to 95.5% changing both number of boosting iteration and weight Trim Beta, and further modifying those variable will decrease accuracy rate.

Changing them separately reduces accuracy. For J48graft, a class for generating a grafted (pruned or unpruned) C4.5 decision tree, unpruned tree gives accuracy of 95.3%. With respect to LAD Tree, before the number of boosting iterations reach 14, accuracy increases with its increase, when it is 14 or 15, accuracy is 95.3%. Furthering increasing will reduce the accuracy again.

LMT builds classification trees with logistic regression functions at the leaves. Using heuristic that avoids cross-validating the number of Logit-Boost iterations at every node produces better outcome. Changing the minimum number of instances at which a node is considered for splitting does not make a difference. Number of boosting iteration chosen based on cross validation is better. With 0.2 as the beta value for weight trimming in LogitBoost, splitting criterion based on the residuals of LogitBoost and AIC used to determine when to stop LogitBoost iterations, the accuracy reaches to 97.33%. Snapshot 1 is the running information of LMT and Snapshot 2 is the outcome.

Snapshot 1:

```
=== Run information ===

Scheme:weka.classifiers.trees.LMT -R -I -1 -M 15 -W 0.2 -A
Relation:     iris.txt
Instances:    150
Attributes:   5
              SL
              SW
              PL
              PW
              class
Test mode:10-fold cross-validation
```

Snapshot 2:

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         146               97.3333 %
Incorrectly Classified Instances         4                2.6667 %
Kappa statistic                          0.96
Mean absolute error                      0.0395
Root mean squared error                  0.1243
Relative absolute error                  8.8907 %
Root relative squared error             26.3771 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          1        Iris_setosa
                 0.96      0.02       0.96      0.96      0.96       0.996    Iris_versicolor
                 0.96      0.02       0.96      0.96      0.96       0.996    Iris_virginica
Weighted Avg.    0.973     0.013      0.973     0.973     0.973      0.998

=== Confusion Matrix ===

  a  b  c   <-- classified as
 50  0  0 |  a = Iris_setosa
  0 48  2 |  b = Iris_versicolor
  0  2 48 |  c = Iris_virginica
```

Dataset: Congressional Voting Data

Table 2.Performance Summary for Congressional Voting Dataset

Approach	Conjunctive Rule	DecisionTable	DJNB	JRip	NNge	OneR
Accuracy	95.6%	95.6%	95.6%	96.3%	94.7%	95.6%
Approach	PART	FT	LMT	Prism	Ridor	ADTree
Accuracy	96.3%	97.0%	96.7%	92.8%	96.1%	96.6%
Approach	BFTree	DecisionStamp	SimpleCart			
Accuracy	95.6%	95.6%	95.6%			

Congressional Voting dataset has 435 rows where there is also no missing value in Congressional Voting dataset and all attributes only contains three values(y, n, w). Conjunctive Rule, Decision Table, DJNB, JRip, NNge, OneR, PART, Prism and Ridor are all approaches under rules. The rest 6 are under trees. Because all attributes are discrete, all these models can be used to train them. Prism is an algorithm based on ID3, uses a different induction strategy than ID3 to induce rules which are modular ("PRISM: An algorithm for inducing modular rules" by Jadzia Cendrowska). There is not parameter we can choose for PRISM and the accuracy of PRISM is only 92.8% which is the lowest one among 15 approaches. So PRISM is not a good approach to train this kind of dataset.

FT is a classifier for building 'Functional trees' and it contains logistic regression function at the inner nodes. With default parameter, it can reach accuracy of 96.7%. Parameter minNumInstances has default value 15 and when I put minNumInstances down, accuracy will go down along with it. When I put minNumInstances up to 120, accuracy can reach 97.0% and accuracy remains 97.0% when I continually put this parameter up but when minNumInstance equals 200, accuracy will suddenly drop down to 61%. When I change the modelType, accuracy remain the same for FT and FTLeaves but accuracy will drop down to 96.7% when I choose FTInner for medelType. Then I try to change the parameter numBoostingIterations, accuracy will

go down. After changing useAIC from true to false, it will not make any difference for accuracy. When I adjust weightTrimBeta, accuracy also goes down. Therefore, minNumInstances is the only parameter which can pull up accuracy. Snapshot 1 is the running information of FT and Snapshot 2 is the outcome.

Snapshot 1:

```
=== Run information ===

Scheme:weka.classifiers.trees.FT -I 15 -F 0 -M 150 -W 0.0
Relation:     house-votes-84.txt
Instances:    435
Attributes:   17
              Party
              handicapped_infants
              water_project_cost_sharing
              adoption_of_the_budget_resolution
              physician_fee_freeze
              el_salvador_aid
              religious_groups_in_schools
              nti_satellite_test_ban
              aid_to_nicaraguan_contras
              mx_missile
              immigration
              synfuels_corporation_cutback
              education_spending
              superfund_right_to_sue
              crime
              duty_free_exports
              export_administration_act_sa
Test mode:10-fold cross-validation
```

Snapshot 2:

```
=== Summary ===

Correctly Classified Instances         422               97.0115 %
Incorrectly Classified Instances        13                2.9885 %
Kappa statistic                          0.9372
Mean absolute error                      0.0501
Root mean squared error                  0.1585
Relative absolute error                 10.566  %
Root relative squared error             32.5476 %
Total Number of Instances              435

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.97      0.03       0.953     0.97      0.962      0.995    republican
                 0.97      0.03       0.981     0.97      0.976      0.995    democrat
Weighted Avg.    0.97      0.03       0.97      0.97      0.97       0.995

=== Confusion Matrix ===

   a   b   <-- classified as
 163   5 |   a = republican
   8 259 |   b = democrat
```