Julia Tzu-Ya Weng U07487022

### ***Data Mining for Scientific Application***

Course No. CSE-4v770

***Laboratory Assignment I:***



***To download additional .arff data sets go to:***

http://repository.seasr.org/Datasets/UCI/arff/

or search the Internet for .arff files required

* What's the difference between a "training set" and a "test set"?

|  |
| --- |
| Training set is data that has a set of attributes and sometimes may contain a class is used to train a model. Independent unseen data that is used to evaluate the performance of the trained model is the test set. |

* Why might a pruned decision tree that doesn't fit the data so well be better than an un-pruned one?

|  |
| --- |
| An un-pruned tree may fit the training data set too tightly such that the tree becomes very big and has many branches with rules specific to the training data such that accuracy looks very good. This, however, causes overfitting. When test data is applied, accuracy is actually lower!  Pruning trims off the branches by removing nodes that give little classifying power and, therefore, have less of an impact on the accuracy but reduce the complexity of the tree. In this way, the overfitting problem is reduced and when a test data set is applied, the model performance is improved compared to an un-pruned tree. |

* What's the first thing that 1R does when making a rule based on a numeric attribute?

|  |
| --- |
| 1R uses a discretization method to convert numeric attributes to nominal attributes. When performing discretization, we use a predefined set of intervals and partition the attribute values according to those interval values. |

* How does 1R avoid overfitting when making a rule based on an enumerated and/or numeric attribute?

|  |
| --- |
| When discretizing a numeric attribute, 1R tries to avoid overfitting by setting a minimum number of examples for the majority class in each partition. |

* What is the difference between Attribute, Instance and Training set?

|  |
| --- |
| The three terms are different terms that are related to each other in that:  An instance is an individual, independent example of the concept, or “the thing”,  to be learned.  The feature of values that measure and characterize aspects of an instance are  the attributes.  We try to learn “the thing” or concept bying given the learner a set of instances, and these are called the training set. |

* What is the difference between ID3 and C4.5?

|  |
| --- |
| ID3 builds decision trees in a top-down fashion and utilizes a greedy algorithm. Each node corresponds to an attribute that maximizes information gain and minimizes entropy, and each leaf branches off from that node to a possible value of that attribute. At each node, data are tested based on that attribute and the result is used to split the data into the leaves. The process repeats until all instances belonging to the same leaf belong to the same class, such that data cannot be split any further.  C4.5 is an improved version of ID3 and handles things that were overlooked in ID3, such as missing values, noise, continuous attribute value ranges, pruning (bottom-up), and applying different weights to the attributes of the training data. |

1. Use the following learning schemes to analyze the iris data (in iris.arff):

|  |  |
| --- | --- |
| OneR | - weka.classifiers.OneR |
| Decision table | - weka.classifiers.DecisionTable -R |
| C4.5 | - weka.classifiers.j48.J48 |

* Do the decisions made by the classifiers make sense to you? Why? Why not? Did you expect a different outcome?

|  |
| --- |
| The decision table uses 3 rules in its classification of the iris data: if not in the majority class, search forward, and stop after 5 node expansion. Though the majority class rule makes sense, for what attribute it is basing its search on is a bit unclear to me.  OneR bases its classification on different width intervals of the petalwidth: <0.8, <1.75, >=1.75. I didn’t expect such a simple rule to generate good classification, but the result is surprisingly good.  J48 rules make more sense to me. It starts with evaluating petalwith <= 0.6, > 0.6. For those with petalwidth >0.6, it looks at whether the petal width is >1.7 or <=1.7. If it is <= 1.7, it will look at whether petallength <= 4.9 or >4.9. If it is the latter, it will then look at petalwidth again to see if it is <=1.5 or >1.5. It seems to include potential exceptions to some of the rules. |

* What can you say about the accuracy of these classifiers? What would happen if we try classifying Iris that has not been used for training?

|  |
| --- |
| OneR has 92% correctly classified instances, decision table, 92.6667%, and J48, 96%.  However, because the models were built on the training set, if we use an unseen test dataset, the accuracy would likely go down. However, because J48 uses rules that are more specific to the nature of the training set, I suspect that accuracy may be decreased the most for the J48 model. |

* How did each one of the methods perform? What metric would you use to compare them and why?

|  |
| --- |
| In terms of % correctly classified, J48 (96%) > decision table (92.6667%) > OneR (92%). In terms of accuracy in area under ROC curve, J48 > decision table = OneR.  The following is the weight average results of the other performance parameters of all three models. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | TP Rate | FP Rate | Precision | Recall | F-  Measure | MCC | ROC Area | PRC Area |
| OneR | 0.920 | 0.040 | 0.920 | 0.920 | 0.920 | 0.880 | 0.940 | 0.876 |
| Decision Table | 0.900 | 0.058 | 0.921 | 0.900 | 0.899 | 0.852 | 0.940 | 0.874 |
| J48 | 0.960 | 0.020 | 0.960 | 0.960 | 0.960 | 0.940 | 0.968 | 0.924 |

|  |
| --- |
| All models have a kappa statistic of greater than 0, indicating that they are doing better than by chance. For initial comparison, I want to first focus on the F-measure, ROC area, and root mean squared error.  Reasons:  -F-measure: help me consider precision and recall together  -ROC area: gives me an idea about model accuracy  -absolute mean error: tells me about the magnitude of error/distance between the model’s prediction and the actual data points. It can also give me an idea of how closely the model is fitted to the data, how flexible the model is and if overfitting may be a problem.  Using the experiment environment to run statistical analysis on the three models on the above mentioned parameters, the models did not differ significantly in terms of F-measure and ROC area. Therefore, the models are not significantly different from each other in terms of precision, recall, or accuracy, even though J48 seems to have a little bit higher performance. However, OneR has a significantly higher mean absolute error compared to the other two models. OneR may be a poorer model on this particular dataset; however, it can also mean it’s more flexible.  In general, J48 outperformed the other two models slightly.  Note that the most successfully correctly classified class is iris-setosa class by all three models. Iris-setosa may be the most common class in this data set. It might be worth our while to go back and examine if we have an unbalanced data set and how that may affect the accuracy of our models. |

1. Use the following learning schemes to analyze the bolts data (bolts.arff without the TIME attribute):

|  |  |
| --- | --- |
| Decision Tree | - weka.classifiers.j48.J48 |
| Decision table | - weka.classifiers.DecisionTable -R |
| Linear regression | - weka.classifiers.LinearRegression |
| M5' | - weka.classifiers.M5' |

* The dataset describes the time needed by a machine to produce and count 20 bolts. (More details about the data can be found in the file containing the dataset.) Tip: what is the most important attribute in the Decision Tree?

|  |
| --- |
| T20BOLT is the most important attribute because it will be used as the class attribute |

* Analyze the data. What adjustments to which specific attributes would have the greatest effect on the time to count 20 bolts?

|  |
| --- |
| TBOLT20 was discretized for decision table and J48. All four classifiers agree that adjusting SPEED1 will have the greatest effect on time. It is the most important or the only rule in all four classifiers. |

* According to each classifier, how would you adjust the machine to get the shortest time to count 20 bolts?

|  |
| --- |
| J48:   * set SPEED1 at 2 or 4   Decision Table:   * SPEED1 <= 4   Linear regression:   * T20BOLT = 8.6906 \* SPEED1 + -3.1687 \* SENS + 21.3528 * decreaseing SPEED1 decreases T20BOLT   M5P:   * if SPEED1 <= 5, follow model LM1 to adjust T20BOLT:   T20BOLT = 1.0449 \* SPEED1 - 0.1752 \* TOTAL - 1.2187 \* SENS + 27.6057   * if SPEED2 > 5, follow model LM2 to adjust T20BOLT:   T20BOLT = 4.2051 \* SPEED1 + 0.754 \* TOTAL - 1.5333 \* SENS +  26.7562   * both models indicate that decreasing SPEED1 decreases T20BOLT   In general, SPEED1 should be set at <= 4 for reduced T20BOLT. NOTE that both linear regression and M5P models suggest that increasing SENS may decrease  T20BOLT. The documentation, however, says that SENS should not affect time. Is there a difference in energy consumption or does SENS really have no impact on time? Also, J48 suggests that in order to get reduced T20BOLT for SPEED1 = 6, TOTAL has to be 10, which does not make sense as we want number of bolts to count to be 20. Therefore, we need to consult the company and this data should be reviewed and re-analyzed to determine if TOTAL and SENS should be included as attributes. |

1. Produce a model for both Weather and Weather.nominal data sets. Which method(s) did you use? What did the tree(s) look like?

|  |
| --- |
| I used four classifiers with 3-fold cross validation: decision table, oneR, Naive Bayes, and J48 for both weather and weather.nominal datasets.  Overall, Naive Bayes the best classifier for both the nominal and numeric dataset.  Decision table:  The poorest among all the classifiers tested. TP, FP, precision, recall and F-measure are 0 for class play = no, indicating that all instances are classified as play = yes. Both kappa statistic and MCC have a value of 0. The classifier is no better than guessing randomly. Simplifying the data by categorizing humidity and temperature attributes did not improve the performance.  OneR:  For the model built from the numeric data, MCC (0.189) and kappa statistics (0.186) are low. 5 out of 14 instances were incorrectly classified. TP, FP, precision, recall and F-measure for play = no is much lower than play = yes, indicating that classification performance for play = no is low. Area under ROC curve is 0.589. Classifier is slightly better than tossing a coin.  Using the nominal data to build the OneR based model, performance measures are more balanced, but accuracy is still low. 5 out of 14 instances were still incorrectly classified. Converting humidity and temperature attributes to nominal values did not improve the overall performance.  J48:  Similar to OneR, given the numeric data set, J48 tends to misclassify play = no to play = yes. There are 5 play = no instances, and 4 were misclassified as play = yes. Even though area under ROC curve is above 0.6, MCC is only 0.122, and TP, FP, precision, recall and F-measure for play = no is much lower compared to play = yes. Again, using humidity and temperature attributes as nominal values makes the performance measures more balanced, but 5 out of 14 instances were still incorrectly classified. Even though the rules in J48 are easy to understand, in this case, J48 is not the most reliable classifier.  The following figure shows the difference in the J48 tree generated on the nominal and the numeric dataset. The rule for splitting at the humidity node is different.    Figure 1. Decision tree generated from the weather.nominal dataset    Figure 2. Decision tree generated from the weather.numeric dataset  Naive Bayes:  This is the best out of the four classifiers I have tested; 11 out of 14 instances are correctly classified. FP rate is low for both play classes. Area under ROC is above 0.8. Converting the humidity and temperature attributes to nominal improves the classification performance a little bit: F-measure, area under ROC, kappa statistics, MCC, are all improved; mean absolute error is reduced. However, the increase in performance is in the expense of increased FP rate in predicting play = yes. We could use collect more data and test the classifiers again to help determine whether using the nominal humidity and temperature attributes is helpful. |