CSCI4030U: Big Data Analytics
Lab06

Syed Naqvi
100590852

April 8, 2024

1.

(a)

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     lymphography
Instances:    148
Attributes:   19
              lymphatics
              block_of_affere
              bl_of_lymph_c
              bl_of_lymph_s
              by_pass
              extravasates
              regeneration_of
              early_uptake_in
              lym_nodes_dimin
              lym_nodes_enlar
              changes_in_lym
              defect_in_node
              changes_in_node
              changes_in_stru
              special_forms
              dislocation_of
              exclusion_of_no
              no_of_nodes_in
              class
Test mode:    evaluate on training data
```

```
=== Classifier model (full training set) ===

J48 pruned tree
------------------

lym_nodes_dimin <= 1
|   changes_in_node = no
|   |   defect_in_node = no: normal (3.0/1.0)
|   |   defect_in_node = lacunar: malign_lymph (2.0)
|   |   defect_in_node = lac_margin: normal (0.0)
|   |   defect_in_node = lac_central: normal (0.0)
|   changes_in_node = lacunar
|   |   exclusion_of_no = no: metastases (10.0/1.0)
|   |   exclusion_of_no = yes
|   |   |   special_forms = no: metastases (3.0/1.0)
|   |   |   special_forms = chalices
|   |   |   |   lym_nodes_enlar <= 2: malign_lymph (3.0)
|   |   |   |   lym_nodes_enlar > 2: metastases (2.0)
|   |   |   special_forms = vesicles: malign_lymph (19.0/1.0)
|   changes_in_node = lac_margin
|   |   block_of_affere = no
|   |   |   extravasates = no
|   |   |   |   lymphatics = normal: metastases (0.0)
|   |   |   |   lymphatics = arched
|   |   |   |   |   early_uptake_in = no: metastases (5.0/1.0)
|   |   |   |   |   early_uptake_in = yes: malign_lymph (4.0/1.0)
|   |   |   |   lymphatics = deformed: metastases (5.0)
|   |   |   |   lymphatics = displaced: malign_lymph (1.0)
|   |   |   extravasates = yes: malign_lymph (4.0)
|   |   block_of_affere = yes: metastases (56.0/3.0)
|   changes_in_node = lac_central
|   |   no_of_nodes_in <= 1
|   |   |   block_of_affere = no: malign_lymph (3.0)
|   |   |   block_of_affere = yes: metastases (2.0)
|   |   no_of_nodes_in > 1: malign_lymph (20.0)
lym_nodes_dimin > 1
|   by_pass = no: metastases (2.0/1.0)
|   by_pass = yes: fibrosis (4.0)

Number of Leaves  :     21

Size of the tree :      34


Time taken to build model: 0.01 seconds
```

```
=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances         138               93.2432 %
Incorrectly Classified Instances        10                6.7568 %
Kappa statistic                          0.8722
Mean absolute error                      0.0545
Root mean squared error                  0.1651
Relative absolute error                 20.3659 %
Root relative squared error             45.3684 %
Total Number of Instances              148

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               1.000    0.007    0.667      1.000    0.800      0.814    0.997     0.667     normal
               0.963    0.104    0.918      0.963    0.940      0.864    0.966     0.952     metastases
               0.885    0.023    0.964      0.885    0.923      0.875    0.967     0.958     malign_lymph
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     fibrosis
Weighted Avg.  0.932    0.067    0.936      0.932    0.933      0.872    0.968     0.952

=== Confusion Matrix ===

  a  b  c  d   <-- classified as
  2  0  0  0 |  a = normal
  1 78  2  0 |  b = metastases
  0  7 54  0 |  c = malign_lymph
  0  0  0  4 |  d = fibrosis
```

The weka J48 algorithm is an open source version of the C4.5 algorithm which is essentially a statistical classification method. It creates a decision tree by using the principals of information entropy to recursively partitioning the dataset based on the attributes with the highest information gain. Once the tree has been constructed, it is pruned by removing branches that have little to no contribution to the classification accuracy as a result of minial information gain. Once pruning has completed, the algorithm then converts the decision tree into a set of if-then rules to simplify representation.

(b)

```
=== Classifier model (full training set) ===

JRIP rules:
===========

(lymphatics = normal) => class=normal (2.0/0.0)
(lym_nodes_dimin >= 2) and (by_pass = yes) => class=fibrosis (4.0/0.0)
(no_of_nodes_in >= 3) and (special_forms = vesicles) => class=malign_lymph (41.0/5.0)
(block_of_affere = no) and (extravasates = yes) => class=malign_lymph (8.0/0.0)
(changes_in_node = lac_central) => class=malign_lymph (8.0/2.0)
 => class=metastases (85.0/11.0)

Number of Rules : 6


Time taken to build model: 0.02 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances         130               87.8378 %
Incorrectly Classified Instances        18               12.1622 %
Kappa statistic                          0.7688
Mean absolute error                      0.1045
Root mean squared error                  0.2286
Relative absolute error                 39.0327 %
Root relative squared error             62.8081 %
Total Number of Instances              148
```

```
=== Run information ===

Scheme:       weka.classifiers.rules.JRip -F 3 -N 2.0 -O 2 -S 1
Relation:     lymphography
Instances:    148
Attributes:   19
              lymphatics
              block_of_affere
              bl_of_lymph_c
              bl_of_lymph_s
              by_pass
              extravasates
              regeneration_of
              early_uptake_in
              lym_nodes_dimin
              lym_nodes_enlar
              changes_in_lym
              defect_in_node
              changes_in_node
              changes_in_stru
              special_forms
              dislocation_of
              exclusion_of_no
              no_of_nodes_in
              class
Test mode:    evaluate on training data
```

```
=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     normal
               0.914    0.164    0.871      0.914    0.892      0.754    0.888     0.853     metastases
               0.820    0.080    0.877      0.820    0.847      0.748    0.885     0.825     malign_lymph
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     fibrosis
Weighted Avg.  0.878    0.123    0.879      0.878    0.878      0.762    0.891     0.847

=== Confusion Matrix ===

  a  b  c  d   <-- classified as
  2  0  0  0 |  a = normal
  0 74  7  0 |  b = metastases
  0 11 50  0 |  c = malign_lymph
  0  0  0  4 |  d = fibrosis
```

The weka JRip algorithm is an implementation of the RIPPER (Repeated Incremental Pruning to Produce Error Reduction) algorithm. This is a rule based classification method that iteratively constructs a set of if-then rules to classify data. Initially the rules are grown by adding conditions to minimize the Error on the training set followed by a pruning process where rules are removed to eliminate overfitting and improve generalization. This method is repeated for each class treating multi-class classification as a series of binary problems that are optimized by removing or replacing rules in order to improve overall accuracy. The final model consists of a sequence of easy to understand rules that can be used to classify new instances.

2.

(a) **C4.5 (weka.classifier.trees.J48)**
Classification Accuracy: 97.2222%
Confusion matrix:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 204   0 |   a = 0
  12 216 |   b = 1
```

Just as in the previous part of this lab, the J48 algorithm creates a decision tree partitioning on attributes with the highest information gain. Once a decision tree has been created, it is pruned to remove any branches with nominal contribution to classification accuracy. The final tree is then used to classify new instances.

(b) **RIPPER (weka.classifier.rules.JRip)**
Classification Accuracy: 90.2778%
Confusion matrix:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 186  18 |   a = 0
  24 204 |   b = 1
```

This algoritm first generates a set of if-then rules seeking to minimize classification error by adding as many conditions as possible. It then prunes the rules to reduce over-fitting and increase generalization. This process is repeated for each class and the result is a straight-forward set of rules that can be used to classify new instances.

(c) **k-Nearest Neighbor (weka.classifiers.lazy.IBk)**
Classification Accuracy: 87.5%
Confusion matrix:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 192  12 |   a = 0
  42 186 |   b = 1
```

This algorithm stores the training dataset in memory and calcualtes the distance (could be euclidean or manhattan or something else) between a new instance and its 'k' closest neighbours. The instance is then assigned the class most common amongst its neighbours.

(d) **Naive Bayesian Classification (weka.classifiers.bayes.NaiveBayes)**
Classification Accuracy: 97.2222%
Confusion matrix:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 204   0 |   a = 0
  12 216 |   b = 1
```

This algorithm makes strong independence assumptions between features and uses Bayes' theorem to determine the probability of a class given a set of input features. It then uses the features of an unseen instance to determine the most likely class.

(e) **Neural Networks (weka.classifiers.functions.MultilayerPerceptron)**
Classification Accuracy: 93.5185%
Confusion matrix:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 204   0 |   a = 0
  28 200 |   b = 1
```

This algorithm is an implementation of a feedforward ariticial neural network where each neuron in one layer connects to every neuron in the next layer. Features in the training data are passed through the the layers and processed via weighted connections and activation functions to produce a prediction. The prediction is then compared to the actual output and then back propagated through the network to adjust the weights in reverse and minimize error in the case of a new instance.