CSE634: Classifying Type of Rocks on Bakary Dataset

Professor Anita Wasilewska
Team Number: 12
Team Members:
Pragesh Jagnani - 112045076
Nikhil Navadiya - 112046312
Sanjay Mathew Thomas - 112026123
Supreeth Narasimhaswamy - 112007225

Table of Contents

1. Project Description

Data Mining is a process used to extract unknown patterns from large volumes of data. It can be divided into various steps such as Data Preprocessing, Data Mining (Proper), and Interpretation. Most commonly referred to Knowledge Discovery in Databases (KDD) process. The aim of this project is to build two types of classifiers i.e. descriptive and non-descriptive classifier to classify different Rock types. For descriptive classifier, decision tree based classification technique is used, and for the non-descriptive classifier, neural network based classification technique is used. Data used to build these classifiers can be found here. There are 98 records, 48 attributes, and 6 classes namely R. Carbonatees AND R. Carbonatees impures, Pyrate, Charcopyrite, Galene, Spahlerite, and Sediments terrigenes in the data. We used the software, Weka to build these classifiers and obtain the results. This report is divided into the following sections: Data Preparation which lists out the steps followed to prepare or clean the data, Data Preprocessing which lists out the steps followed to discretize and normalize the data, Data Mining Proper/Methods which provides details about the classifiers, and finally Interpretation/Conclusion.

2. Data Preparation

To prepare the data for the classification process, we first convert the XLS file into CSV format so that it can be imported into Weka. Then we applied the following steps in order to clean the data:

- Removed the second row which was an empty row to avoid it being used in calculations.
- The last row of the data set which consists of the repetition of column names was removed.
- Removed the “Echantillon” column which consists of sample id for data samples. This column was removed since it is unique for every row and the classifiers wouldn’t benefit from this column.
- One of the values in the “Li” column was “<0.3” which could not be parsed by Weka. Hence, we had changed it to “0.3”.
- There was a total of 8 distinct classes in the data, we renamed the incorrectly spelled classes to reduce the number of distinct classes to 6.
- Removed the attributes/columns which consists of more than 70% of missing values. Namely, Co(85%), Mo(89%), Cd(70%), As(72%).
- Missing values were filled using the ReplaceMissingValues filter in Weka. This will replace all the missing values for each attribute using its mean value from among all the training records.

After completing the above steps, we employed Data Preprocessing techniques.

3. Data Preprocessing

After completing the Data Preparation stage, we have the data relevant to our experiment, which we shall refer to as ‘PD’. Our next task is to prepare the data for the decision tree classifier. We employ two different data discretization techniques on PD namely, Equal width binning and Equal frequency binning. We used the Discretize filter in Weka for equal width binning, setting the number of bins to 4. We chose 4 as our number of bins after analyzing the values of all attributes. This resulted in the creation of our first data set named PD1. Then, we applied Equal frequency binning to PD using the PKIDiscretize filter in Weka. This resulted in the second data set named PD2. Both PD1 and PD2 will be used for the decision tree based classifier.

Data Normalization is an important step to use neural network based classification. So, we used Normalize filter in Weka over Project data (PD) to make it suitable for neural network based classifier.

4. Data Mining Methods (DM Proper)

4.1 Decision Tree

Algorithm

We used J48 for decision tree classification. J48 is an open source Java implementation of C4.5 which is an algorithm to generate a decision tree developed by Ross Quinlan. J48 also allow us to select if we want to prune tree or not.

Validation method

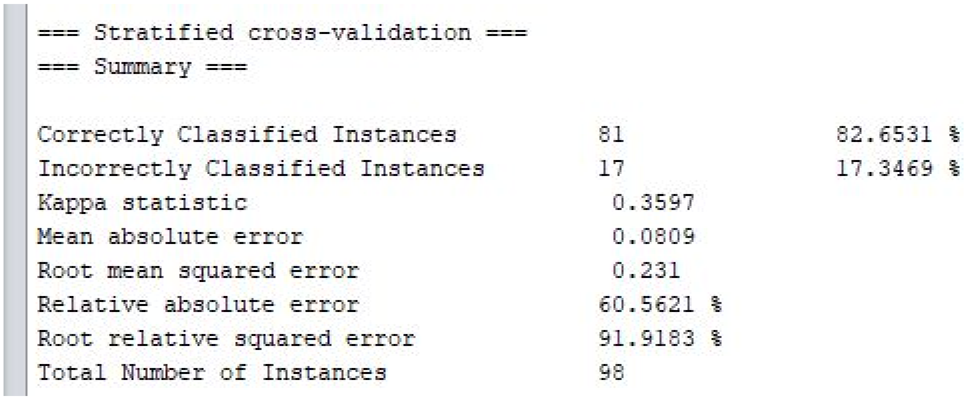
In all experiment for the decision tree, we used LOOCV (leave one out cross validation) for testing purpose since we had less amount of training data. When we have less amount of data, LOOCV is the preferred choice for validation of the model.

4.1.1 Dataset PD1

4.1.1.1 Experiment 1 – Full Classification (Learning)

Parameters used for best accuracy

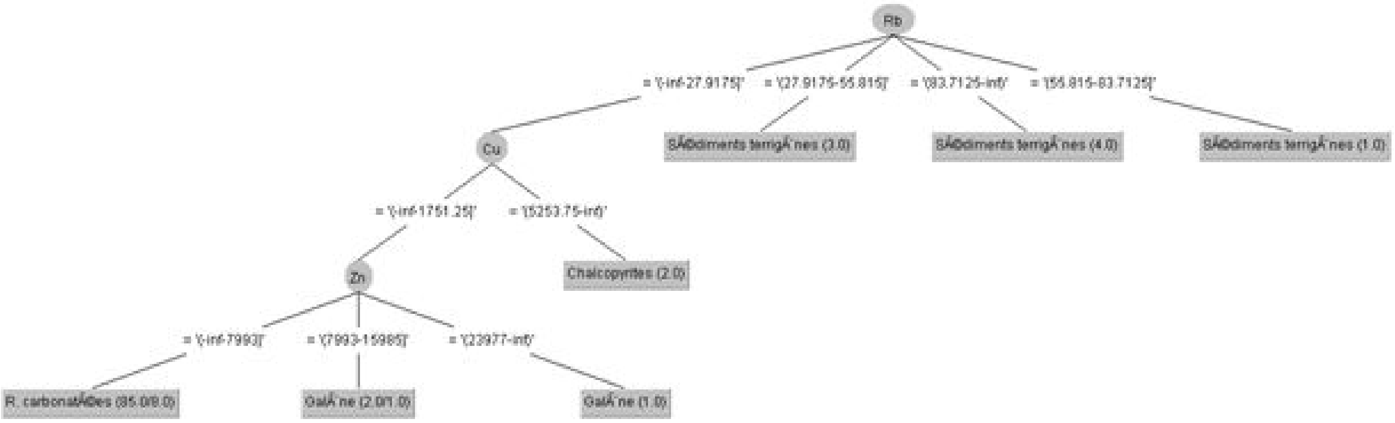
Pruned = true
Confidence factor = 0.25
Minimum number of instances per leaf = 2
Accuracy: 82.6531%



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

Correctly Classified Instances          81               82.6531 %
Incorrectly Classified Instances        17               17.3469 %
Kappa statistic                          0.3597
Mean absolute error                      0.0809
Root mean squared error                  0.231
Relative absolute error                 60.5621 %
Root relative squared error             91.9183 %
Total Number of Instances               98
```

Tree model



Discriminant rules learned

Rule 1: If Rb is less than 27.9175 and Cu is less than 1751.25 and Zn is less than 7993 then type of rock is R. carbonatées

Rule 2: If Rb is less than 27.9175 and Cu is less than 1751.25 and Zn is greater than 7993 then type of rock is Galène
Rule 3: If Rb is less than 27.9175 and Cu is greater than 5253.75 then type of rock is Chalcopyrites
Rule 4: If Rb is greater than 27.9175 then type of rock is Sédiments terrigènes

4.1.1.2 Experiment 2 – Contrast learning

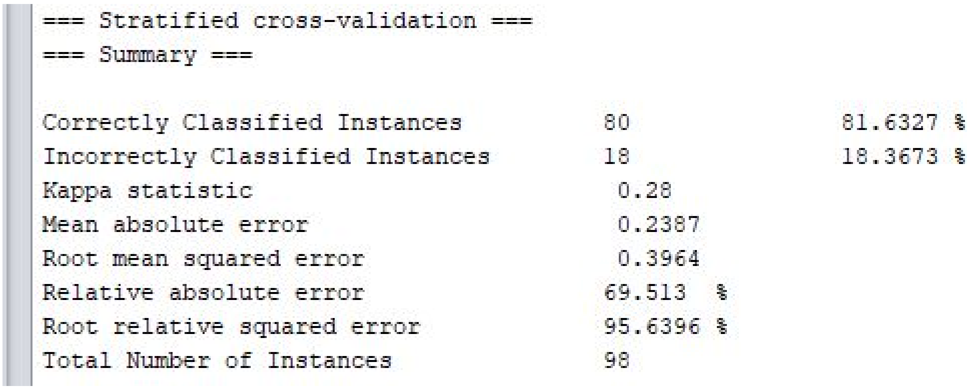
Two classes

1. C1 (R. carbonatées)
2. noC1 – (all other classes)

Parameters used for best accuracy

Pruned = true
Confidence factor = 0.25
Minimum number of instances per leaf = 2

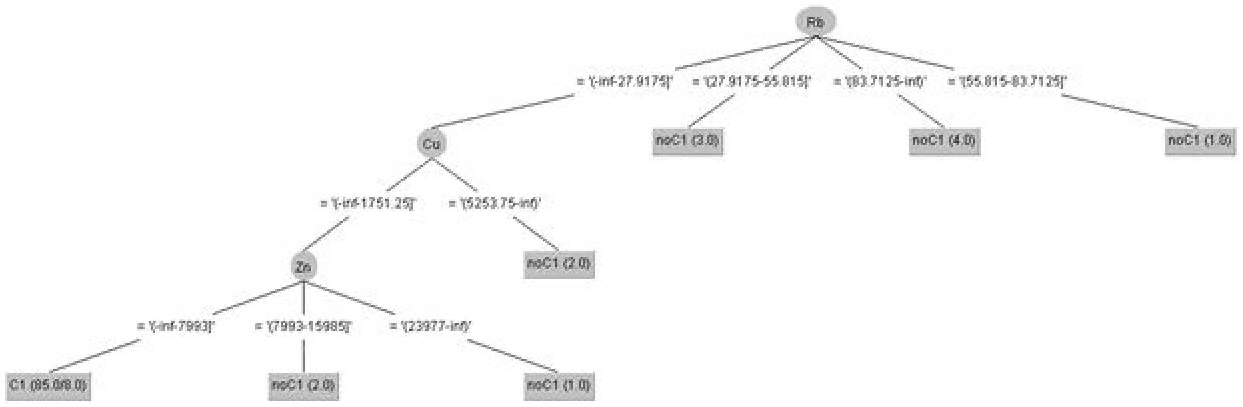
Accuracy: 81.6327%



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

Correctly Classified Instances          80               81.6327 %
Incorrectly Classified Instances        18               18.3673 %
Kappa statistic                          0.28
Mean absolute error                      0.2387
Root mean squared error                  0.3964
Relative absolute error                 69.513  %
Root relative squared error             95.6396 %
Total Number of Instances               98
```

Tree model



Discriminant rules learned

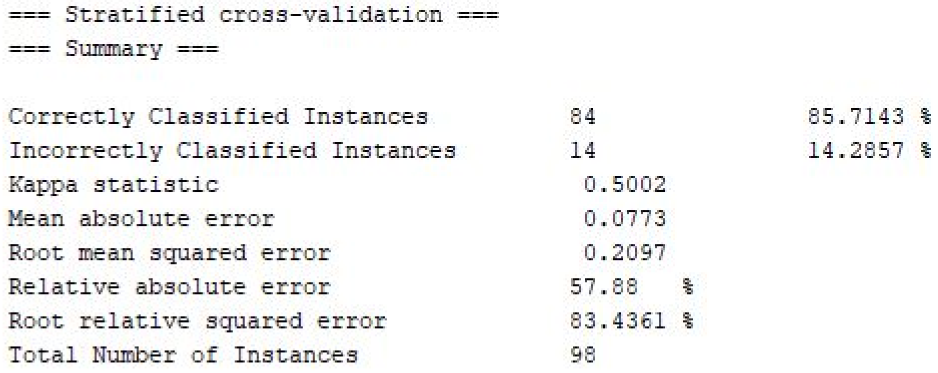
Rule 1: If Rb is less than 27.9175 and Cu is less than 1751.25 and Zn is less than 7993 then type of rock is C1
Rule 2: If Rb is less than 27.9175 and Cu is less than 1751.25 and Zn is greater than 7993 then type of rock is noC1
Rule 3: If Rb is less than 27.9175 and Cu is greater than 5253.75 then type of rock is noC1
Rule 4: If Rb is greater than 27.9175 then type of rock is noC1

4.1.1.3 Experiment 3 (only important attributes)

4.1.1.3.1 Exp 1 - full classification (learning)

Parameters used for best accuracy

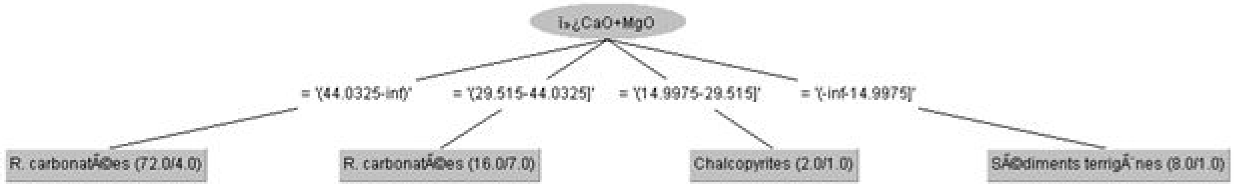
Pruned = false
Minimum number of instances per leaf = 2
Accuracy: 85.7143%



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

Correctly Classified Instances          84               85.7143 %
Incorrectly Classified Instances        14               14.2857 %
Kappa statistic                          0.5002
Mean absolute error                      0.0773
Root mean squared error                  0.2097
Relative absolute error                 57.88    %
Root relative squared error             83.4361 %
Total Number of Instances               98
```

Tree Model



Discriminant rules learned

Rule 1: If CaO+MgO is less than 14.9975 then type of rock is Sédiments terrigènes
Rule 2: If CaO+MgO is greater than 14.9975 and CaO+MgO is less than 29.515 then type **of** rock is Chalcopyrites
Rule 3: If CaO+MgO is greater than 29.515 then type of rock is R. carbonatées

4.1.1.3.2 Exp. 2 – Contrast Learning

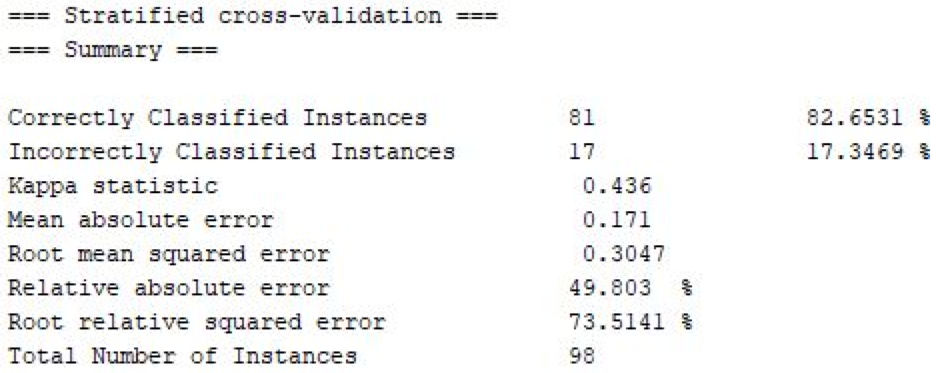
Two classes

1. C1 (R. carbonatées)
2. noC1 (all other classes)

Parameters used for best accuracy

Pruned = false
Minimum number of instances per leaf = 2

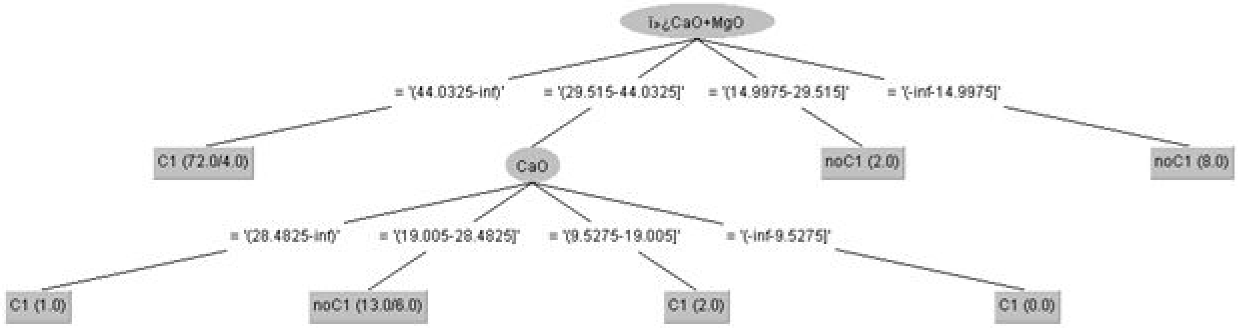
Accuracy: 82.6531%



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

Correctly Classified Instances          81               82.6531 %
Incorrectly Classified Instances        17               17.3469 %
Kappa statistic                          0.436
Mean absolute error                      0.171
Root mean squared error                  0.3047
Relative absolute error                 49.803  %
Root relative squared error             73.5141 %
Total Number of Instances               98
```

Tree Model



Discriminant rules learned

Rule 1: If CaO+MgO is less than 29.515 then type of rock is noC1
Rule 2: If CaO+MgO is greater than 29.515 and CaO+MgO is less than 44.0325 and CaO is less than 19.005 then type of rock is C1
Rule 3: If CaO+MgO is greater than 29.515 and CaO+MgO is less than 44.0325 and CaO is greater than 19.005 and less than 28.4825 then type of rock is noC1
Rule 4: If CaO+MgO is greater than 29.515 and CaO+MgO is less than 44.0325 and CaO is greater than 28.4825 then type of rock is C1
Rule 5: If CaO+MgO is greater than 44.0325 then type of rock is C1

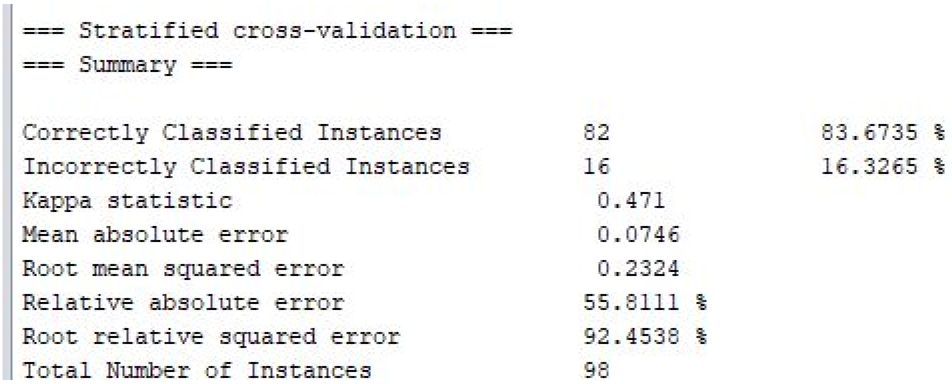
4.1.2 Dataset PD2

4.1.2.1 Experiment 1 – Full Classification (Learning)

Parameters used for best accuracy

Pruned = true
Confidence factor = 0.6
Minimum number of instances per leaf = 9

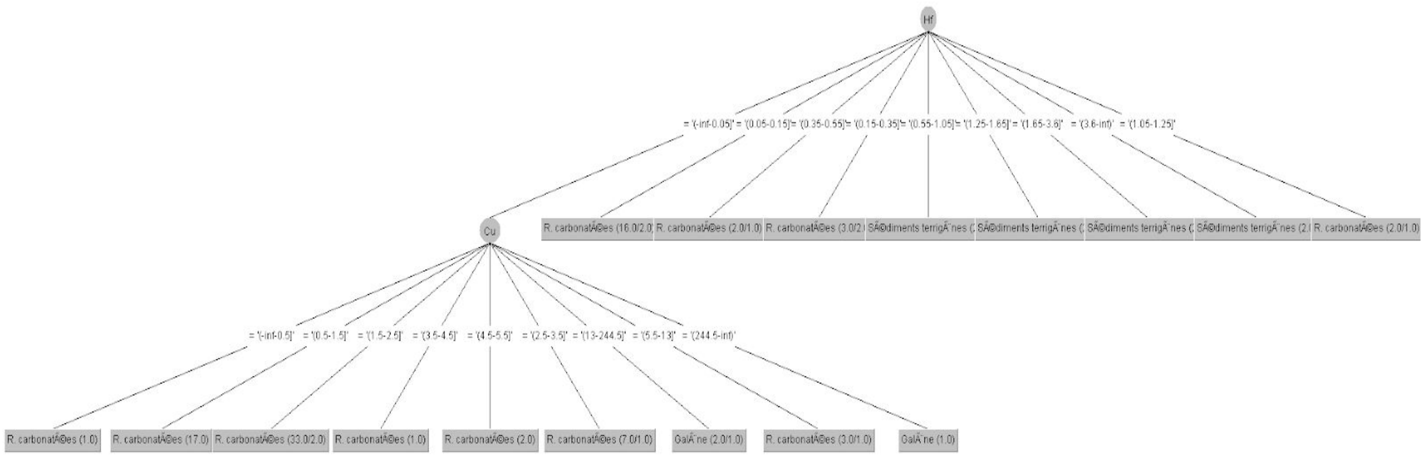
Accuracy: 83.6735%



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

Correctly Classified Instances          82               83.6735 %
Incorrectly Classified Instances        16               16.3265 %
Kappa statistic                          0.471
Mean absolute error                      0.0746
Root mean squared error                  0.2324
Relative absolute error                 55.8111 %
Root relative squared error             92.4538 %
Total Number of Instances               98
```

Tree model



Discriminant rules learned

Rule 1: If Hf is less than 0.05 and Cu is less than 13 then type of rock is R. carbonatées
Rule 2: If Hf is less than 0.05 and Cu is greater than 13 then type of rock is Galène
Rule 3: If Hf is greater than 0.05 and Hf is less than 0.55 then type of rock is R. carbonatées
Rule 4: If Hf is greater than 0.55 and Hf is less than 1.05 then type of rock is Sédiments terrigènes
Rule 5: If Hf is greater than 1.05 and Hf is less than 1.25 then type of rock is R. carbonatées
Rule 6: If Hf is greater than 1.25 then type of rock is Sédiments terrigènes

4.1.2.2 Experiment 2 – Contrast Learning

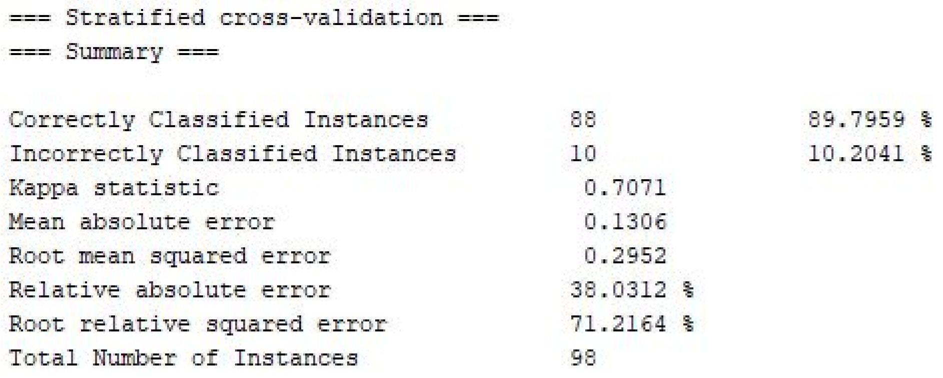
Two classes

1. C1 (R. carbonatées)
2. noC1 – (all other classes)

Parameters used for best accuracy

Pruned = true
Confidence factor = 0.6
Minimum number of instances per leaf = 4

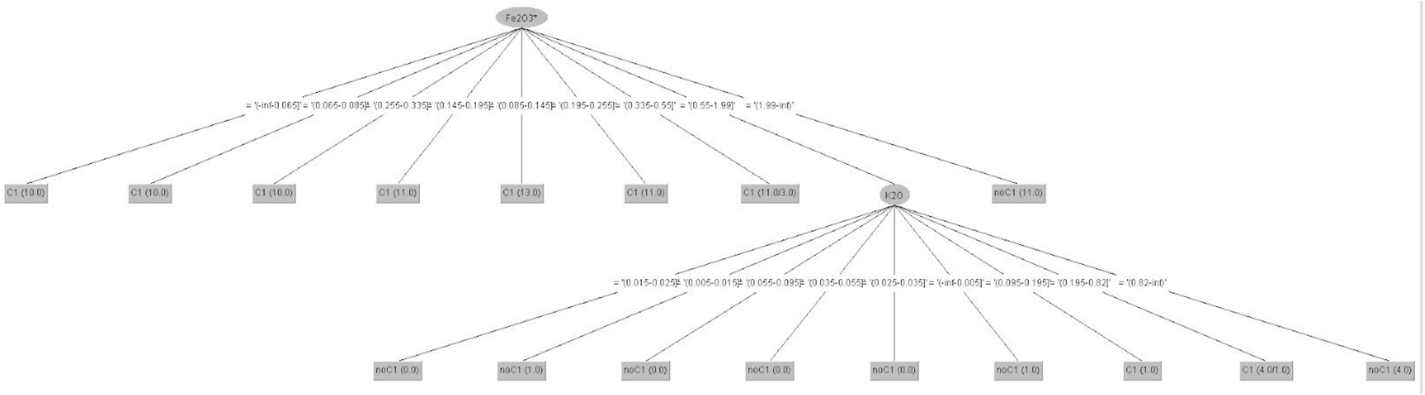
Accuracy: 89.7959%



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

Correctly Classified Instances          88               89.7959 %
Incorrectly Classified Instances        10               10.2041 %
Kappa statistic                          0.7071
Mean absolute error                      0.1306
Root mean squared error                  0.2952
Relative absolute error                 38.0312 %
Root relative squared error             71.2164 %
Total Number of Instances               98
```

Tree Model



Discriminant rules learned

Rule 1: If Fe2O3 is less than 0.55 then type of rock is C1

Rule 2: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and K2O is less than 0.095 then type of rock is noC1

Rule 3: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and K2O is greater than 0.095 and K2O is less than 0.82 then type of rock is C1

Rule 4: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and K2O is greater than 0.82 then type of rock is noC1

Rule 5: If Fe2O3 is greater than 1.99 then type of rock is noC1

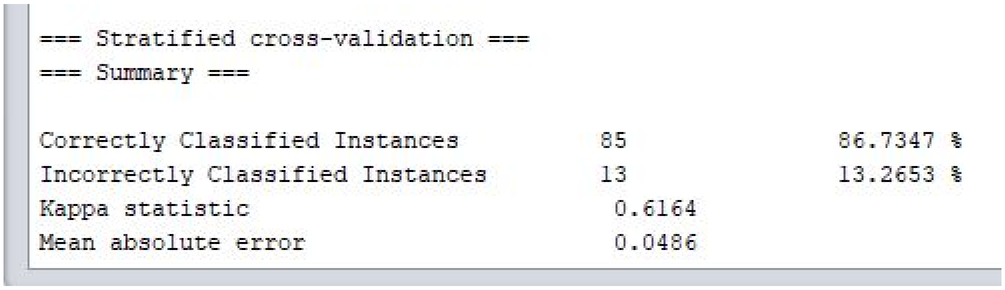
4.1.2.3 Experiment 3 (only important attributes)

4.1.2.3.1 Exp1 – Full Classification (Learning)

Parameters used for best accuracy

Pruned = true
Confidence factor = 0.8
Minimum number of instances per leaf = 4

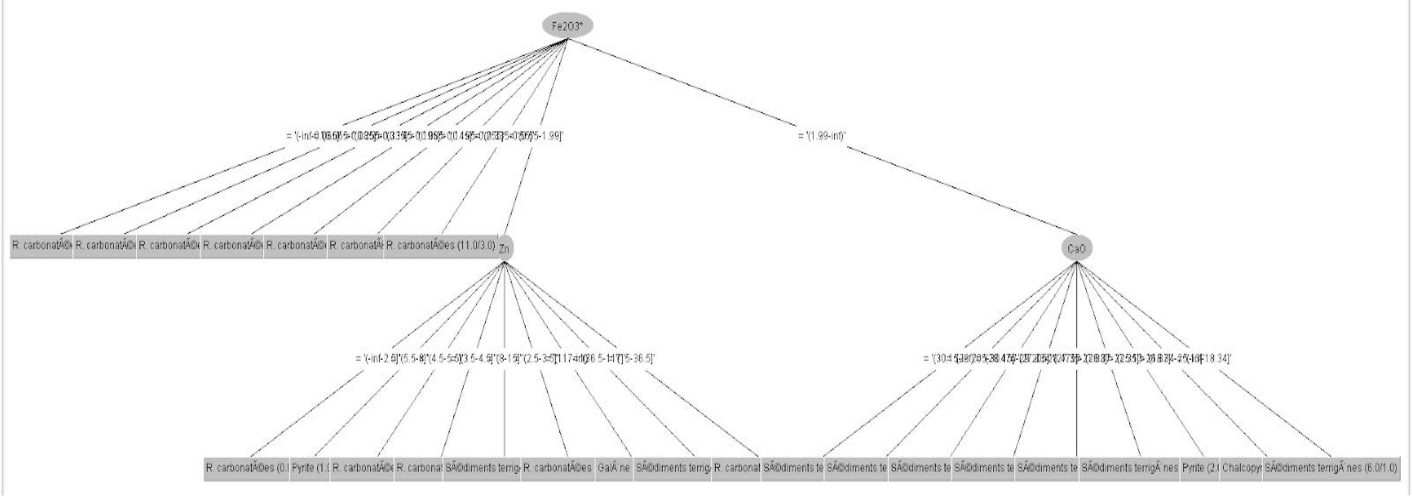
Accuracy: 86.7347%



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

Correctly Classified Instances          85               86.7347 %
Incorrectly Classified Instances        13               13.2653 %
Kappa statistic                          0.6164
Mean absolute error                      0.0486
```

Tree Model



Discriminant rules learned

Rule 1: If Fe2O3 is less than 0.55 then type of rock is R. carbonatées
Rule 2: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and Zn is less than 5.5 then type of rock is R. carbonatées
Rule 3: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and Zn is greater than 5.5 and Zn is less than 8 then type of rock is Pyrite
Rule 4: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and Zn is greater than 8 and Zn is less than 15 then type of rock is Sédiments terrigènes
Rule 5: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and Zn is greater than 15 and Zn is less than 36.5 then type of rock is R. carbonatées

Rule 6: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and Zn is greater than 36.5 and Zn is less than 117 then type of rock is Sédiments terrigènes
Rule 7: If Fe2O3 is greater than 0.55 and Fe2O3 is less than 1.99 and Zn is greater than 117 then type of rock is Galène
Rule 8: If Fe2O3 is greater than 1.99 and CaO is less than 18.34 then type of rock is Sédiments terrigènes
Rule 9: If Fe2O3 is greater than 1.99 and CaO is greater than 18.34 and less than 25.13 then type of rock is Chalcopyrites
Rule 10: If Fe2O3 is greater than 1.99 and CaO is greater than 25.13 and less than 26.87 then type of rock is Pyrite
Rule 11: If Fe2O3 is greater than 1.99 and CaO is greater than 26.87 then type of rock is Sédiments terrigènes

4.1.2.3.2 Exp2 Contrast Learning

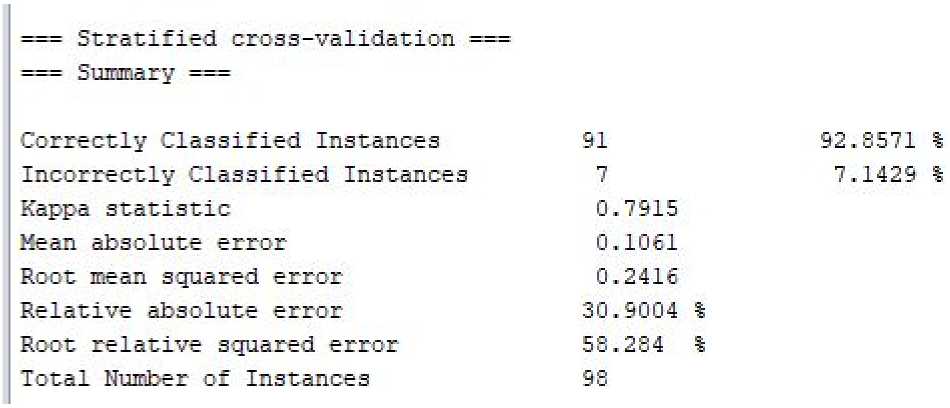
Two classes

1. C1 (R. carbonatées)
2. noC1 – (all other classes)

Parameters used for best accuracy

Pruned = true
Confidence factor = 0.25
Minimum number of instances per leaf = 2

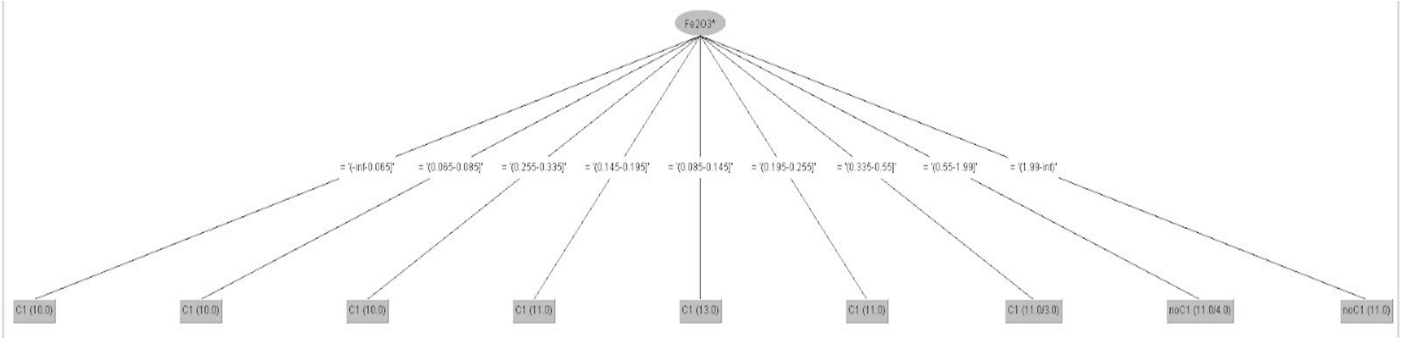
Accuracy: 92.8571%



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

Correctly Classified Instances          91               92.8571 %
Incorrectly Classified Instances         7                7.1429 %
Kappa statistic                          0.7915
Mean absolute error                      0.1061
Root mean squared error                  0.2416
Relative absolute error                 30.9004 %
Root relative squared error             58.284  %
Total Number of Instances               98
```

Tree Model



Discriminant rules learned

Rule 1: If Fe2O3 is less than 0.55 then type of rock is C1
Rule 2: If Fe2O3 is greater than 0.55 then type of rock is noC1

4.2 Neural Networks

4.2.1 Introduction

A neural network is a set of connected input/output units where each connection has a weight associated with it. During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples.

The neural network architecture that we use here is a Multilayer Perceptron which is a feed-forward neural network consisting of an input layer, an output layer and any positive number of hidden layers. Each layer of the network could contain any number of nodes. The Multilayer Perceptron uses the Backpropagation algorithm for learning the weights of the various connections in the network. It learns by iteratively processing a set of training data and comparing the network's prediction for each record with the value of the class attribute. The error in the final prediction is propagated back and used to modify the weights of the network.

4.2.2 Parameters

The training process for the Multilayer Perceptron also requires us to specify the values of a set of parameters. Based on these values, a given network architecture could learn a completely different set of weights and could thus produce dramatically different predictions. Hence, the values of these parameters need to be tuned to ensure optimal performance of the neural network. We shall now look into some of the primary parameters that we used for our neural network:

a) Learning Rate:
It determines the extent to which weights should be modified based on the error that is propagated back in a given iteration of the Backpropagation algorithm. The learning rate is always a number in the range [0,1]. Specifying a value that is too high for the learning rate could result in our algorithm missing the global minima. But, specifying a learning rate that is too small could make our algorithm take too long to converge and could also get it stuck in an undesirable local minimum. We may also have the learning rate decay as the number of epochs increases. This is to ensure that the algorithm makes smaller steps as it progresses and gets closer to the global minima. This, however, is optional.

b) Momentum:
The momentum is used along with the learning rate to ensure that the algorithm doesn't get stuck in a local minimum. If we have a larger value of momentum, we should keep the learning rate lower as the algorithm would be making large jumps otherwise, which could result in it missing the global minima.

c) Number of training epochs:
One epoch is said to be completed when all the training records have been passed through the neural network and used to update the weights. By running the algorithm over multiple epochs, we could reduce the prediction error even further. Thus, the greater the number of training epochs, the lesser the prediction error would become. However, after a certain point, the reduction in error would be very small. Increasing the number of epochs past this point would yield no advantage and would only increase the training time.

d) Number of hidden layers:
By building a network with more hidden layers, we would be able to learn more complex relations between the attributes and the class attribute. However, care must be taken not to specify too many hidden layers as the network could overfit to the training data and reduce the prediction accuracy when we use the model against a test set.

4.2.3 Experiments

We shall now describe each of the models we had constructed and the results we obtained with each of them.

4.2.3.1 Experiment 1 – Full Classification (Learning)

Input:

We used all 44 attributes in each record as input for our classifier.

Output:

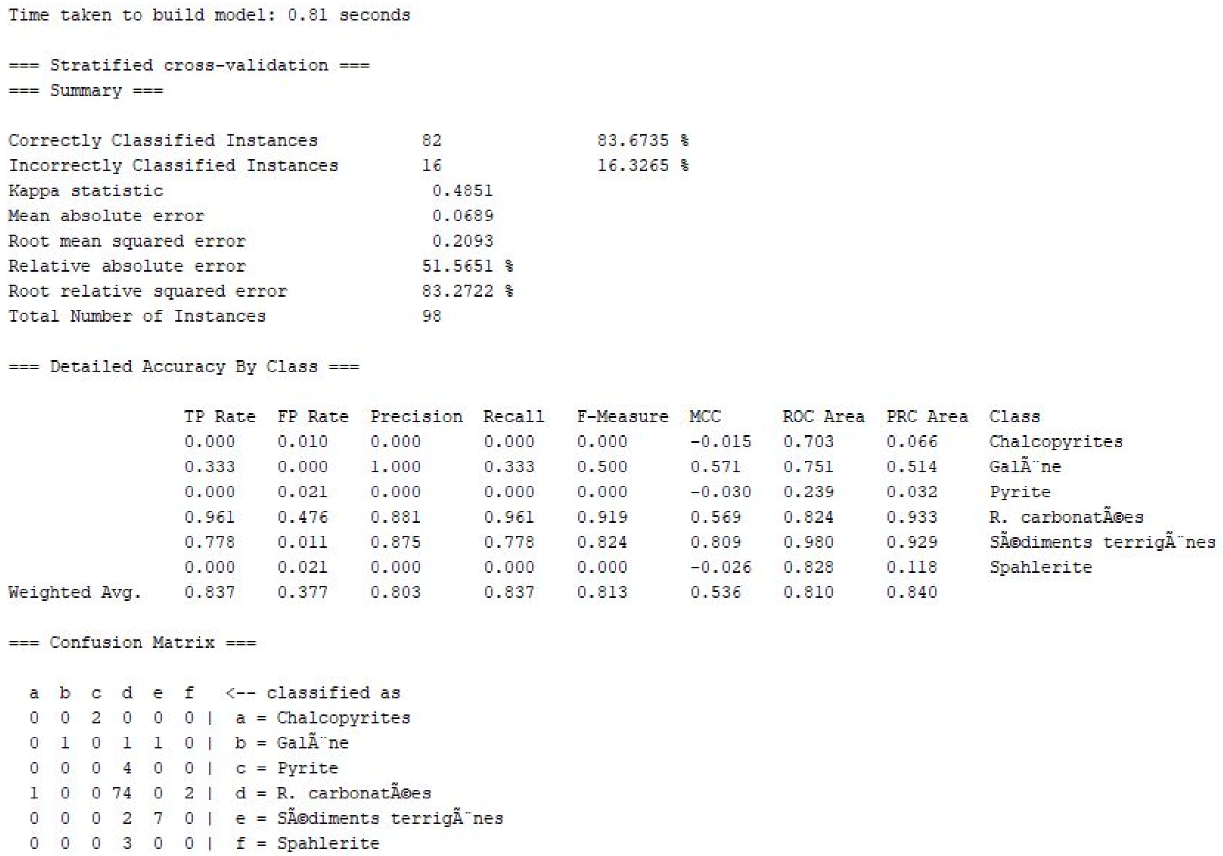
There were six output classes corresponding to each Rock type.

Classifier Parameters:

We had used the following parameters with the Multilayer Perceptron classifier:
Learning Rate: 0.3
Momentum: 0.2
Number of training epochs: 500
Number of hidden layers: Not specified (so that the classifier can learn the optimal number of hidden layers)

Results:

Prediction accuracy: **83.6735%**



```
Time taken to build model: 0.81 seconds

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

Correctly Classified Instances          82               83.6735 %
Incorrectly Classified Instances        16               16.3265 %
Kappa statistic                          0.4851
Mean absolute error                      0.0689
Root mean squared error                  0.2093
Relative absolute error                 51.5651 %
Root relative squared error             83.2722 %
Total Number of Instances               98

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.010    0.000      0.000    0.000      -0.015   0.703     0.066     Chalcopyrites
                 0.333    0.000    1.000      0.333    0.500      0.571    0.751     0.514     GalÃ¨ne
                 0.000    0.021    0.000      0.000    0.000      -0.030   0.239     0.032     Pyrite
                 0.961    0.476    0.881      0.961    0.919      0.569    0.824     0.933     R. carbonatÃ©es
                 0.778    0.011    0.875      0.778    0.824      0.809    0.980     0.929     SÃ©diments terrigÃ¨nes
                 0.000    0.021    0.000      0.000    0.000      -0.026   0.828     0.118     Spahlerite
Weighted Avg.    0.837    0.377    0.803      0.837    0.813      0.536    0.810     0.840

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
  0  0  2  0  0  0 |  a = Chalcopyrites
  0  1  0  1  1  0 |  b = GalÃ¨ne
  0  0  0  4  0  0 |  c = Pyrite
  1  0  0 74  0  2 |  d = R. carbonatÃ©es
  0  0  0  2  7  0 |  e = SÃ©diments terrigÃ¨nes
  0  0  0  3  0  0 |  f = Spahlerite
```

4.2.3.2 Experiment 2 – Contrast Learning

Input:

We used all 44 attributes in each record as input for our classifier.

Output:

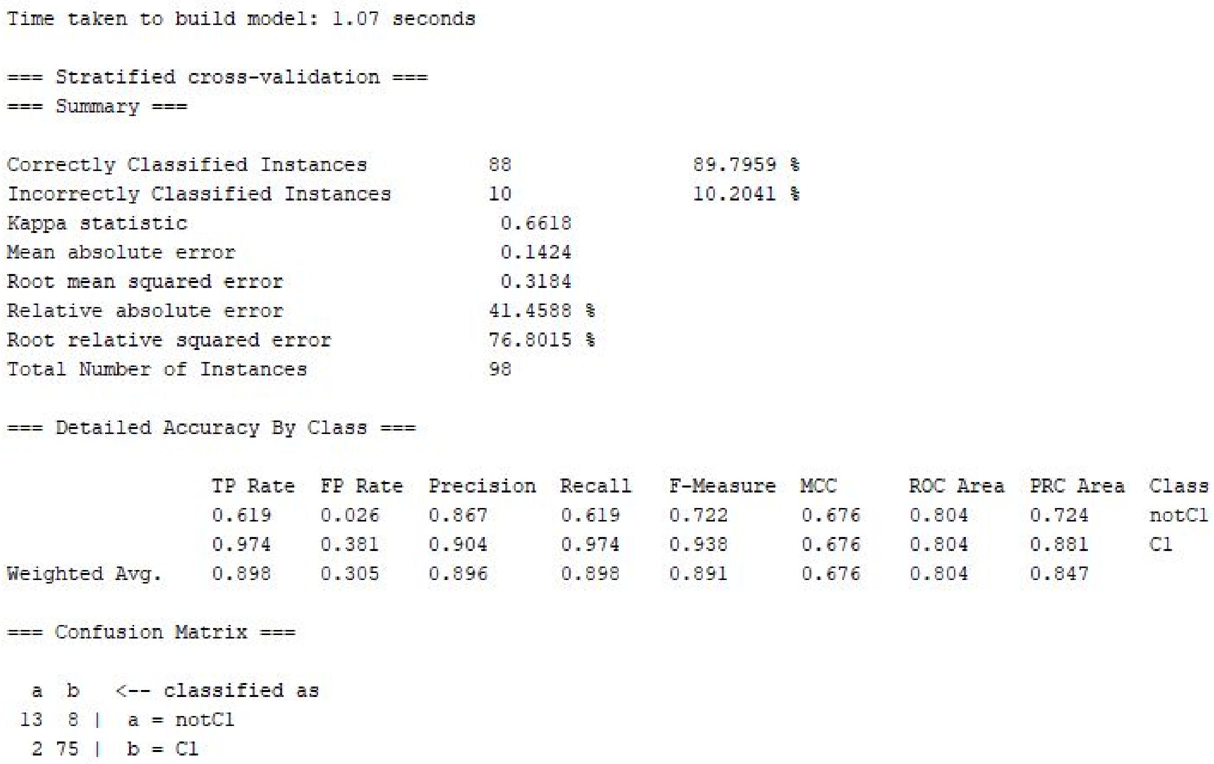
There were only two output classes, class ‘C1’ and ‘notC1’.

Classifier Parameters:

We had used the following parameters with the Multilayer Perceptron classifier:
Learning Rate: 0.3
Momentum: 0.2
Number of training epochs: 500
Number of hidden layers: Not specified

Results:

Prediction accuracy: **89.7959%**



```
Time taken to build model: 1.07 seconds

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

Correctly Classified Instances          88               89.7959 %
Incorrectly Classified Instances        10               10.2041 %
Kappa statistic                          0.6618
Mean absolute error                      0.1424
Root mean squared error                  0.3184
Relative absolute error                 41.4588 %
Root relative squared error             76.8015 %
Total Number of Instances               98

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.619    0.026    0.867      0.619    0.722      0.676    0.804     0.724     notC1
                 0.974    0.381    0.904      0.974    0.938      0.676    0.804     0.881     C1
Weighted Avg.    0.898    0.305    0.896      0.898    0.891      0.676    0.804     0.847

=== Confusion Matrix ===

  a  b   <-- classified as
 13  8 |  a = notC1
  2 75 |  b = C1
```

4.2.3.3 Experiment 3 – Most important attributes

4.2.3.3.1 Full Classification (Learning)

Input:

We only used the 8 most important attributes in each record as input for our classifier.

Output:

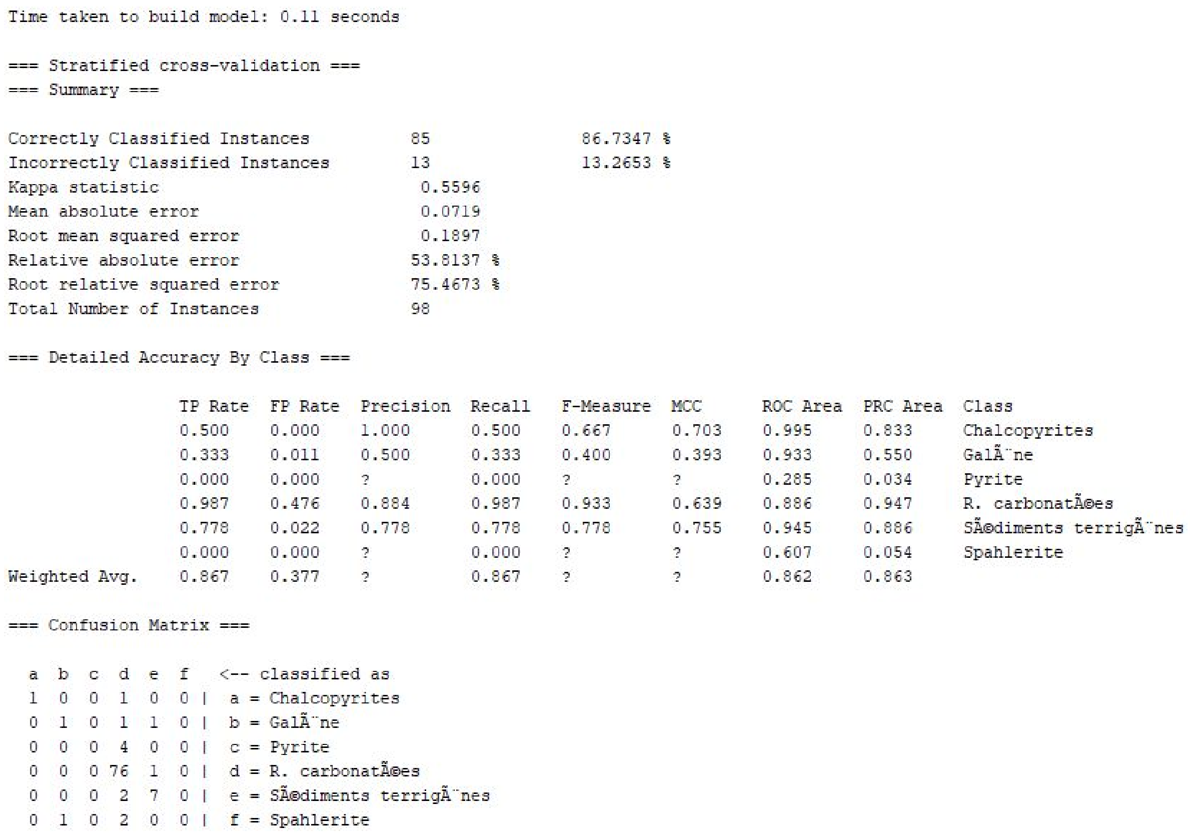
There were six output classes corresponding to each Rock type.

Classifier Parameters:

We had used the following parameters with the Multilayer Perceptron classifier:
Learning Rate: 0.3
Momentum: 0.2
Number of training epochs: 500
Number of hidden layers: Not specified

Results:

Prediction accuracy: **86.7347%**



```
Time taken to build model: 0.11 seconds

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

Correctly Classified Instances          85               86.7347 %
Incorrectly Classified Instances        13               13.2653 %
Kappa statistic                          0.5596
Mean absolute error                      0.0719
Root mean squared error                  0.1897
Relative absolute error                 53.8137 %
Root relative squared error             75.4673 %
Total Number of Instances               98

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.500    0.000    1.000      0.500    0.667      0.703    0.995     0.833     Chalcopyrites
                 0.333    0.011    0.500      0.333    0.400      0.393    0.933     0.550     GalÃ¨ne
                 0.000    0.000    ?          0.000    ?          ?        0.285     0.034     Pyrite
                 0.987    0.476    0.884      0.987    0.933      0.639    0.886     0.947     R. carbonatÃ©es
                 0.778    0.022    0.778      0.778    0.778      0.755    0.945     0.886     SÃ©diments terrigÃ¨nes
                 0.000    0.000    ?          0.000    ?          ?        0.607     0.054     Spahlerite
Weighted Avg.    0.867    0.377    ?          0.867    ?          ?        0.862     0.863

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
  1  0  0  1  0  0 |  a = Chalcopyrites
  0  1  0  1  1  0 |  b = GalÃ¨ne
  0  0  0  4  0  0 |  c = Pyrite
  0  0  0 76  1  0 |  d = R. carbonatÃ©es
  0  0  0  2  7  0 |  e = SÃ©diments terrigÃ¨nes
  0  1  0  2  0  0 |  f = Spahlerite
```

4.2.3.3.2 Contrast Learning

Input:

We only used the 8 most important attributes in each record as input for our classifier.

Output:

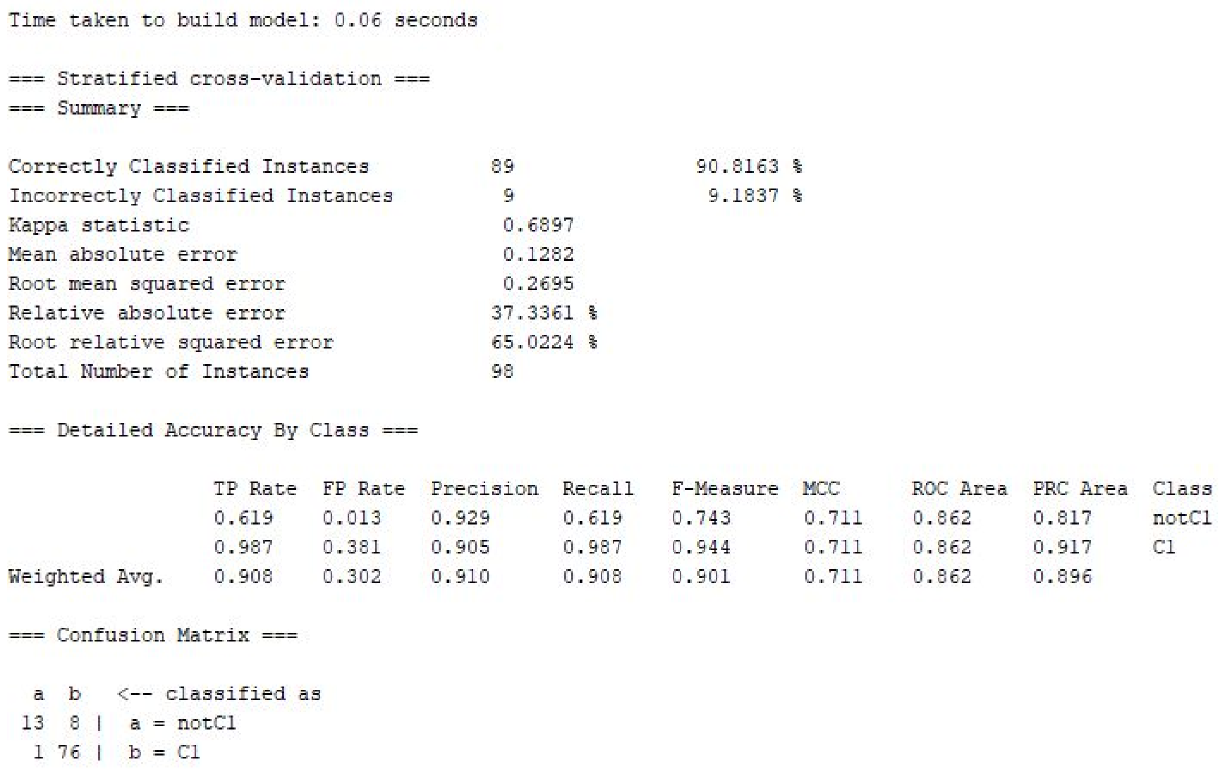
There were only two output classes, class ‘C1’ and ‘notC1’.

Classifier Parameters:

We had used the following parameters with the Multilayer Perceptron classifier:
Learning Rate: 0.3
Momentum: 0.2
Number of training epochs: 500
Number of hidden layers: Not specified

Results:

Prediction accuracy: **90.8163%**



```
Time taken to build model: 0.06 seconds

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

Correctly Classified Instances          89               90.8163 %
Incorrectly Classified Instances         9                9.1837 %
Kappa statistic                          0.6897
Mean absolute error                      0.1282
Root mean squared error                  0.2695
Relative absolute error                 37.3361 %
Root relative squared error             65.0224 %
Total Number of Instances               98

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.619    0.013    0.929      0.619    0.743      0.711    0.862     0.817     notC1
                 0.987    0.381    0.905      0.987    0.944      0.711    0.862     0.917     C1
Weighted Avg.    0.908    0.302    0.910      0.908    0.901      0.711    0.862     0.896

=== Confusion Matrix ===

  a  b   <-- classified as
 13  8 |  a = notC1
  1 76 |  b = C1
```

5. Interpretation and Evaluation

We observe that the highest prediction accuracy was seen with Contrast Learning using only the most important attributes. Among these, the Decision tree classifier had a higher predictive accuracy of **92.8571%** while the Neural Network classifier had an accuracy of **90.8163%**.

The superior performance observed with Contrast Learning could be attributed to the lower number of output classes leading to less ambiguity during classification. Using only the most important attributes also improved the performance as most of the other attributes didn't have much relevance to the class attribute and just contributed to prediction error.

Finally, we believe the decision tree classifier achieved a better predictive accuracy over the neural network classifier because the decision tree classifier operated on nominal/categorical attributes while the neural network operated on numerical attributes. This most likely resulted in the neural network learning noise in the input attribute values thus decreasing its predictive accuracy slightly.

6. Conclusion

We used the Decision Tree and Neural Network classifiers to classify the records in BakaryData. We used both full classification and contrast learning for both these classifiers to produce separate models. We also used both all the available attributes and only the most important attributes to produce different models. We compared the predictive accuracy observed in each of these cases. Based on the evaluated results, we see that the decision tree classifier using contrast learning on only the most important attributes offered the best performance.

7. References

1. Course Website: https://www3.cs.stonybrook.edu/~cse634/
2. Weka Tool: https://www.cs.waikato.ac.nz/~ml/weka/index.html
3. Learning Rate: https://en.wikipedia.org/wiki/Learning_rate