BRAIN TUMOR CLASSIFICATION

Team Members - Archit Jain, Ayush Arora, Siddharth Chauhan

1 INTRODUCTION

The project is built with the aim to automate the process of Classification of Brain Tumors into Malignant and Benign. This could help doctors in diagnosing the patient's treatment plan in a faster way and help save lives.

1.1 Problem Description - Treatment of a brain tumor can't be planned unless it is classified into Benign(Non-Cancerous) or Malignant(Cancerous). With proper preprocessing and extraction of appropriate features from the region of interest (tumor) from the imaged brains can help in the building of a robust machine learning model that can classify the tumor within the test cases within a fraction of seconds vs the biopsy which might take days.

1.2 Motivation - Cancer is the second-highest life taker in the world after cardiovascular diseases. Approximately 80,000 new cases of Brain Tumor are diagnosed within the USA which leads to about 16,000 deaths due to Malignant Brain Tumors yearly[1]. Hence it is the need of the future to automate the process of detection and classification of the tumor. This can be achieved by processing the images generated by imaging techniques such as MRI.

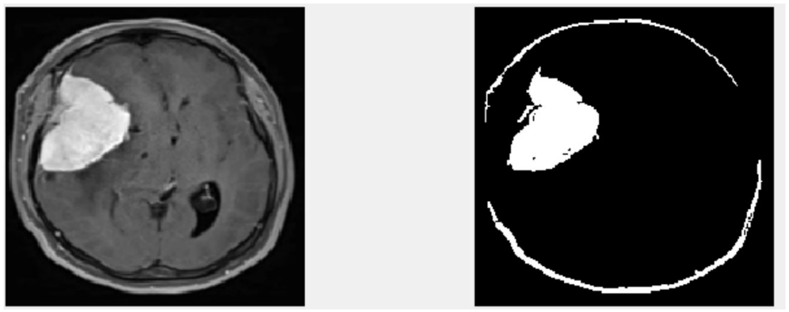
1.3 Brief Description of Report Organization - The report starts with a small Introduction which includes the Problem Description and the Motivation of choosing the problem. Next a brief description of how the data was collected and the steps in understanding the data. Next the preprocessing tasks on the collected images are explained. The preprocessing gives feature sets which are used in the next step of Data Mining in 3 different algorithms Random Forest, k-Nearest Neighbours and the Artificial Neural Networks. A summarization is provided on the various modes the algorithms were run and how the parameters were selected. Then all the performance measures selected are listed with the reasons why they are suitable for the application. Further the experiences of the team on the Logic of Problem are shared followed by the advantages and the disadvantages of the tool. Multiple suggestions on how to make the tool much better are also included. Conclusion summarizes the overall findings of the project. Further Appendix stores the screenshots of the outputs of the models run and the names of the files that can be found within the submission with their purpose. Finally a thankful mention to the work of other people that has helped in the development of the project is given in the references section.

2 DATA EXPLORATION

The data was collected in the form of MRI Images with the file type DICOM. It was downloaded from the Cancer Imaging Archive using their NBIA Retriever. Initially, a search for Brain Tumor Images is run on the Radiology Portal. Following that Brain is selected as the Anatomical site and MRI images are selected and added to the cart from within the collections. Then the collection of DICOM images can be downloaded from the cart as a manifest file. The manifest file, when opened in the NBIA Retriever, extracts out the DICOM Images. These DICOM images can be converted to JPEG using an Online DICOM converter. The frontal lobe view image of the brain is then selected and used as data inputs to a MATLAB code that preprocesses the images and extracts the feature set for the Tumor.

3 METHODOLOGY

3.1 Data Preprocessing - The initial data set consists of about 233 Magnetic Resonated Images which are then processed on the Image Processing program. This program returns a feature set that includes 13 attributes for all the extracted tumors[2]. The feature set includes **Contrast**(the measure of luminance and color that makes it possible to distinguish objects), **Correlation**(the computation of sum of products at each location with respect to a filter being displaced over the image), **Energy**(captures the desired solution and performs descent function over the gradient in order to compute its lowest value), **Homogeneity**(the measure related to the changes in the intensity of a region in an image), **Mean**(the average pixel value), **Standard Deviation**(the measure of the deviation of the pixel values), **Entropy**(The measure of the intensity levels of corresponding states that individual pixels can adapt), **Root Mean Square**(The measure of the differences in the levels of the column vectors), **Variance**(A variance image is an image of the the squares of Standard Deviation), **Smoothness**(The information of pixel pattern generated after the removal of outliers within the pixel data), **Kurtosis**(The measure interpreted in combination of the resolution and noise in the pixel set of the data), **Skewness**(the measure of feature of image that determines it's darkness, glossiness, matte or the lightness) and **Inverse Difference Moment or IDM** (the measure of local homogeneity of the image. The above feature set is calculated using an Image Processing Program based on MATLAB. The program takes a JPEG image as input and segments out the brain tumor and then calculates the features of the tumor image. The figure on the right shows the segmented tumor for the image on the left.



An **Interquartile range filter** is run on the dataset in order to find out the Outliers and the Extreme Values in the dataset. Further, the values that were marked as Outliers or Extreme were processed using the RemovewithValues filter.

3.2 Mining the data

3.2.1 Random Forest: Random-Forest are the decision forest are an ensemble learning method for the classification , regression which is accomplished by constructing a multitude of decision trees at training time. As one of the most powerful algorithms capable of performing both classification and regression tasks[4]. In general more trees in forest, more robust the prediction and thus leads to higher accuracy. In this case an accuracy of 89% is achieved through cross validation technique. As many iterations of decision trees are created the problem of overfitting is solved using these algorithms. As an advantage of the decision tree a high precision rate of 91% is observed as the tree handles various combinations of the attribute to classify without having a problem of overfitting. A recall rate of 86% is also observed which is again a testimony that the algorithm is working in conjunction with the dataset. To introduce randomness a seed value of 1 is used. Bagging is done with 100 iteration and base learners. Random forest algorithms are specifically great at handling classification problems and hence is chosen in this case to classify benign or malign cancers. As the data in this case does not have any noise, hence a greater accuracy is achieved in this case. Hence this becomes a perfect use case to classify problems where random forest becomes a perfect fit. As its disadvantages does not concern this particular data set with no noise and doesn't overfit the problem as well.

3.2.2 K-Nearest Neighbor(IBK) : K-nearest neighbor(KNN) algorithm is one of the popular supervised Machine Learning(ML) algorithm which can be used for both classification and regression predictive problems[3]. KNN is an algorithm that is considered to be both non-parametric and lazy learning, Non-parametric as it does not make any assumptions about the data and lazy as it does not consider any specialized training phase and uses all the data for training. The main idea underlying the use of the KNN algorithm is based on "feature similarity" i.e. it examines how closely the feature of the training set resembles which inturn helps in classifying the data points. It requires loading the training and test data and determining K value i.e the nearest data point.. Further, the distance between test data and training data can be calculated using Euclidean distance. Running the K-Nearest Neighbor algorithm in weka is done using IBK algorithms from the classify section, Setting the initial K value to 20 and applying crossValidate option to true will provide us the optimal value for K in that range. In this case the optimal value for K turns out to be one. Also, selecting the test option as 10-fold cross validation provides us with an accuracy of 81.18% with correctly classified instances as 164 whereas running the same algorithm with percentage split 66% provides an accuracy of 78.26%.

3.2.3 Artificial Neural Network: Artificial Neural Networks(ANN) is a computing system inspired by the neural connection found in the human brain. The main aim of the algorithm is to try to replicate the human brain using various and hence utilizes a technique where it learns using examples instead of being programmed with task-specific rules. The neural network is a collection of connected nodes also known as the perceptrons or the artificial neutrons. These perceptrons have the ability to take inputs either from the outside or from an adjoining perceptron. The main processing task of the ANN occurs using the activation function[5]. The ANN is a three-layered function including the Input Layer, the Hidden Layer, and the Output Layer. All the processing on the inputs occurs at the hidden layer and at the end gives the result in the Output Layer. To run the ANN Algorithm on WEKA the MultilayerPerceptron Classifier is run with a number of hidden layers set as 'a'. Here a = (#attributes + #classes)/ 2 and the seed is set to it's default value zero. The seed is the initialization of the random number generator that is used in the setting of the initial weights. The Learning rate for weight updates is set to 0.3 and the momentum applied to weight updates is 0.2. The algorithm, when run with a 10 folds Cross-Validation on the dataset correctly, classifies 164 correct instances and gives accuracy equivalent to 81% and an accuracy of 71% for Percent Split = 66%.

3.2.4 Model Performance - 3.2.4.1 What Model Performance Measure used?
The performance measures used are : Accuracy, Precision, Recall, F-Measure.
3.2.4.2 Meaning of Model Performance Measure used
Accuracy = Number of correct predictions(TP+TN) / Total number of predictions made(TP+FP+FN+TN).
Accuracy is one of the most intuitive performance measures and is defined as the ratio of total number of correct predictions(True positive and True negative) to the total number of predictions made for a particular dataset. Accuracy is considered to be a good measure in cases where the data being used is nearly balanced or each class contains an equal number of samples. In other cases i.e. for imbalanced classification problems accuracy as a measure is inappropriate as it can result in false sense of high accuracy[7].
Precision = True Positive / (True Positive + False Positive).
As an alternative to accuracy, Precision provides information about the classifier performance with respect to false positives cases. Precision quantifies the correct positive predictions made, in other words precision calculates the positive class predictions actually belonging to positive class.
Recall = True Positive / (True Positive + False Negative)
Recall provides information about classifier performance with respect to false negative cases. Recall quantifies the positive predictions made out of all the positive predictions.It is defined as the number of positive predictions divided by the number of positive class values.
In summary Precision is more appropriate when minimizing false positives and recall is more appropriate when minimizing false negatives is the objective.
F-Measure = 2* 1 / (1/Precision) + (1/Recall)
F-Measure is defined as the Harmonic Mean(HM) of precision and recall. It can take values from [0,1]. The higher the value of F-Measure the better is the performance. Thus, F-Measure provides the combination of both Precision and Recall.
3.2.5 Why choose the selected Model Parameters?
Firstly, Accuracy is selected as the initial measure as the training data being used is balanced/symmetrical around the class labels which in this case is appropriate as it reduces the false sense of high accuracy.Precision is selected as the second performance measure as we want to minimize the false positive cases. Since precision tells us about the proportion of tumors that are classified as malignant are actually malignant.

Moreover, Recall as a third performance measure is taken into consideration as it is not just important to predict the tumor as benign or malignant correctly as in case with precision but also predicting all the cases that have tumor as malignant. Thus, minimizing false negative cases.
Therefore, selecting Precision and recall both as a performance measure insure that we are minimizing both false positive and false negative cases.
Finally, F-Measure as a performance measure has been selected; it represents both Precision and Recall as a single measure and captures both the properties.Also, we are minimizing both false positive and false negative in a single measure. Since F-Measure makes use of harmonic mean instead of arithmetic mean it ensures that extreme values are discounted.

3.2.6 How the performance measures were estimated

1. **Cross-validation**: Cross-validation sampling is used here to evaluate as a dataset is built using limited data. The k value used in while resampling is 10 as the number of instances in the whole dataset is 233. Therefore there are 10 groups of data created from which the whole model is evaluated. Cross validation gives an idea how the model is expected to work and gives a general idea of performance measure which can then be used to compare various algorithms with each other.[6]
2. **Percentage split:** Percentage split is again a resampling method where 66% of data is used for training and remaining 34% is used to perform testing on. The performance measures are calculated against the whole data set while training is done on 66% of data which helps to build an insign of performance of model on the unseen data.
3. **Supplied Test Dataset:** Supplied Test Set is a seperate data file of the format .arff in Weka. This file contains a smaller set of data independent of labels which can be used on the above trained models to determine the label of an unlabeled dataset. The Test Set file contains 10 instances of unknown tumor type. Selecting the best accuracy percentage between the Cross Validation and Percent Split models for all the three algorithms are used to calculate the class labels of the Test Data.

3.2.7 Performance of modes and Table

Algorithm	**Accuracy**	**Precision**	**Recall**	**F-measure**	**ROC**	**Sampling**
RandomForest	89.10%	0.916	0.861	0.888	0.95	Cross-Validation
RandomForest	86.95%	1	0.76	0.866	0.929	Percentage Split
IBK	81.18%	0.812	0.812	0.812	0.799	Cross-Validation
IBK	78.26%	0.787	0.783	0.783	0.775	Percentage Split
Multilayer Perceptron (ANN)	81.18%	0.812	0.812	0.812	0.867	Cross-Validation
Multilayer Perceptron (ANN)	71.01%	0.71	0.71	0.71	0.775	Percentage Split

Analysing the table for the different modes and using the analysis on the various measures mentioned above, looking at the F-measure for the three algorithms we can say that the Random Forests when run with Cross-Validation folds = 10 gives out the best outputs.

4 <u>LOGIC OF PROBLEM</u> - 4.1 Experiences

First step to solving any problem is to understand the problem. Working on the Analytical Thinking procedure of developing the Logic of Problem for the project of Classification of Brain Tumors helped us in understanding the problem and developing a perspective of the problem that could have been easily missed. Further, it helped us collect information about the various aspects of the problem and to structure it in an accessible manner.

Exploring and working on Logic of Problem directed us to the implications as well as the consequences of solving the problem in a particular manner. Working on the problem in stages gave us new issues and hence using the logic of the problem to update our knowledge, procedure and expected outcomes helped provide clarity of the work required to solve the problem.

4.2 PROS: 1. Creating the Logic of Problem helps in better communication about the solution of the problem.

2. It helps think out of the box illicitly.

CONS: 1. It is a little bit time consuming towards the beginning.

2. Isn't collaborative between teams.

3. Misses on the mention of the ethics of solving the problem.

4.3 Any recommendations to add, change or delete any steps:

1. Establish Significance: The importance of the elements in consideration might be different depending on the problem that is being solved. A marking based on the gravity of the elements to solving a problem might help the user understand and come to conclusions in a better way.

2. True Collaboration: Collaborating on a problem with a wider number of users would help bring a bigger perspective towards the solution of the problem. This would also bring people from diverse academic backgrounds together to solve problems efficiently.

3. Ethical Dimension: The goals of critical thinking not only involve developing intellectual ability or rational thinking but also to develop intellectual character. The integration of ethical dimensions and moral responsibilities to the concept of critical thinking can improve the existing model of critical thinking.

5 CONCLUSIONS - 5.1 Compare the results and 5.2 Summary of problem and conclusions drawn

Comparing the results from the different models the following observations have been made:

Random Forest when executed with different test options(percent split and cross validation) performed better as compared to all the other models.

The following patterns was observed while running different models:

While running Random forest, IBK and ANN. The performance of Random forest was consistently higher. Also the accuracy for Random forest was measured as 89% and since the data set user is balanced it does not come under false accuracy. This can be confirmed with the high precision and recall value of Random forest. Finally the high value of F-measure(0.888) confirms that both false positive and false negative are minimised.

It is observed that random forest with 100 bagging iterations and single randomness generating seed result in best performance measure with F-value as 88.8% while using a 10 fold cross validation sampling procedure.

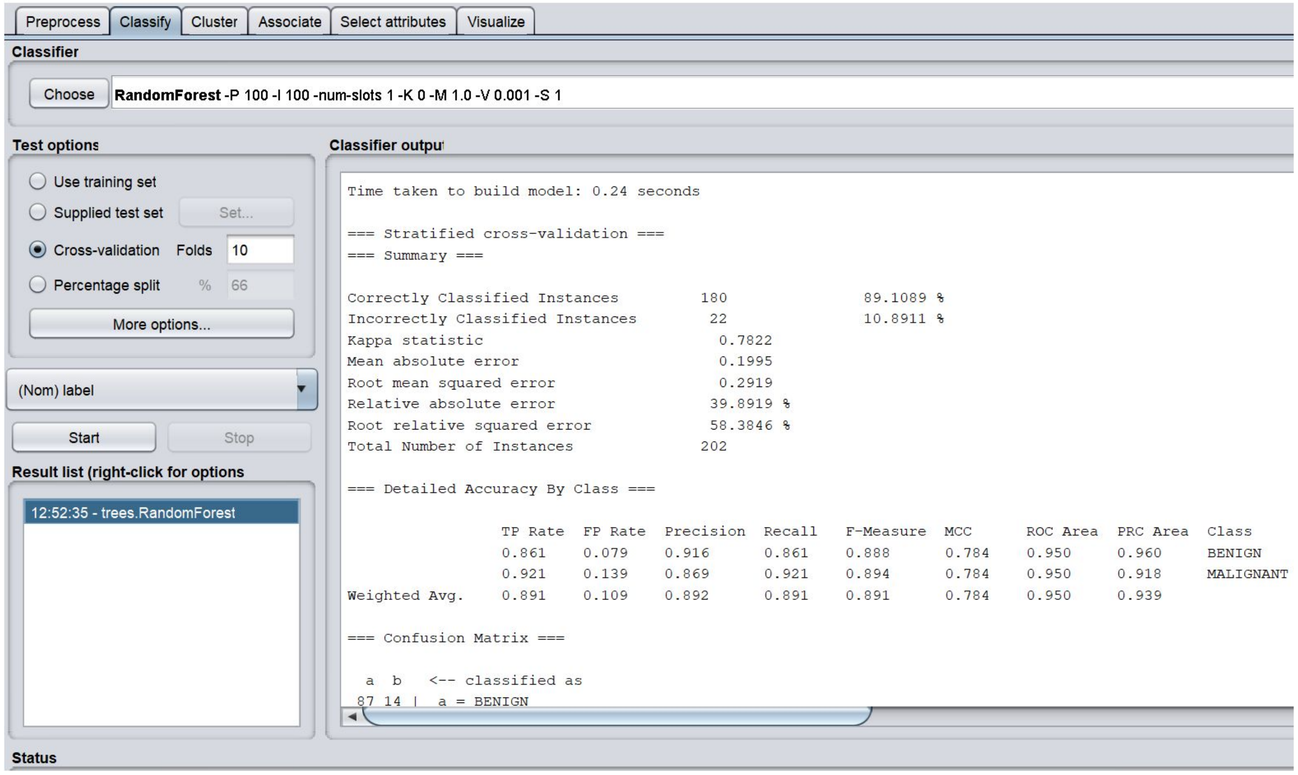
To summarize the problem and conclusion above The main concern for developing an application for the classification of tumors is the results obtained are really sensitive. There is not much scope for error to be present while classification as it would mean life and death. Hence for this case it is essential to minimise the false positive and false negative due to the dramatic impact it can have on people's life. The F measure is chosen as a key performance measure to single down the classification algorithms for the given dataset. As many decision trees are created the problem of overfitting is also resolved with this particular algorithm on brain tumor dataset.

5.3 Future Enhancements : Working on the model further we would like to make use of a bigger dataset which would train the model for a larger range of tumors and help increase the accuracy of the whole model. We would also like to train the model after adding noise filters to add noise so that it could tackle feature sets with noise too. The number of features selected for the image are based on the most commonly used properties of images. The aim would be to expand the features which would have a deeper focus on the region of interest ie the tumor.

6 APPENDIX

RANDOM FOREST OUTPUTS

The output from RandomForest Classifier on the dataset. Number of folds in Cross-Validation = 10



Classifier: RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

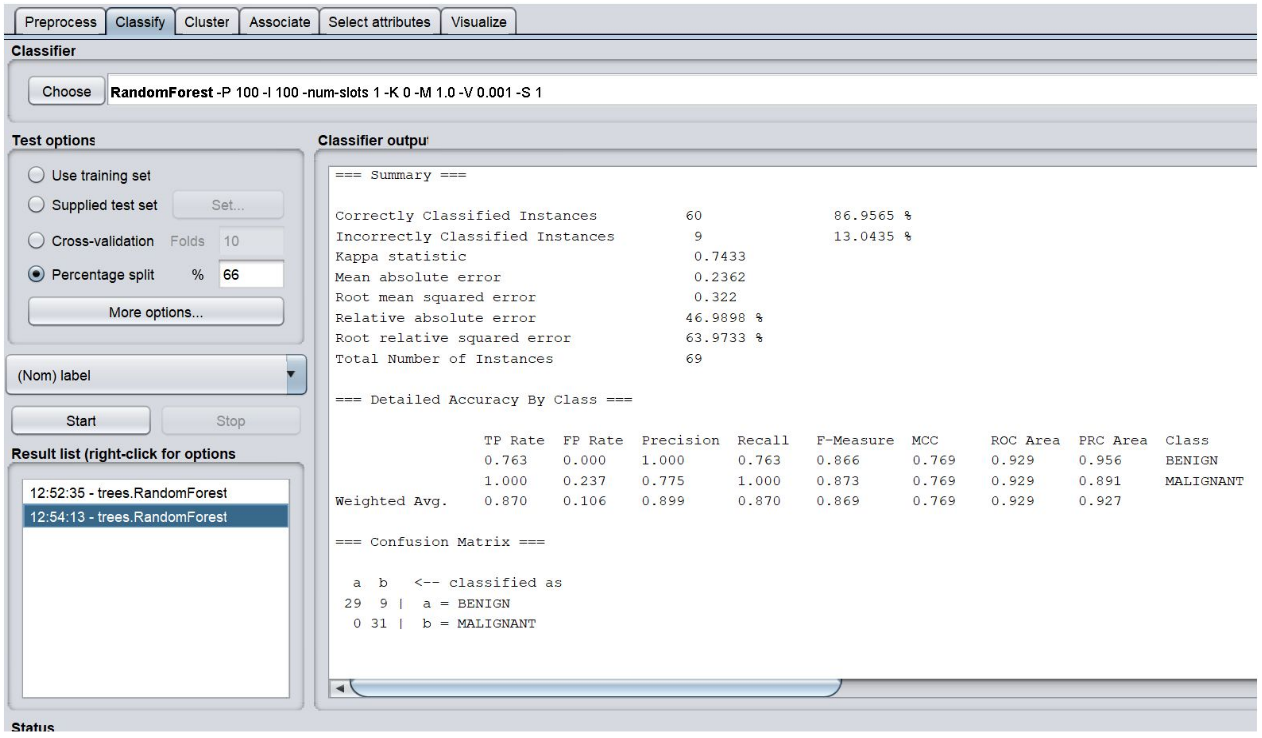
Test options: Cross-validation Folds 10

(Nom) label

Result list: 12:52:35 - trees.RandomForest

```
Time taken to build model: 0.24 seconds

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

Correctly Classified Instances         180               89.1089 %
Incorrectly Classified Instances        22               10.8911 %
Kappa statistic                          0.7822
Mean absolute error                      0.1995
Root mean squared error                  0.2919
Relative absolute error                 39.8919 %
Root relative squared error             58.3846 %
Total Number of Instances              202

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.861    0.079    0.916      0.861    0.888      0.784    0.950     0.960     BENIGN
                 0.921    0.139    0.869      0.921    0.894      0.784    0.950     0.918     MALIGNANT
Weighted Avg.    0.891    0.109    0.892      0.891    0.891      0.784    0.950     0.939

=== Confusion Matrix ===

  a  b   <-- classified as
 87 14 |  a = BENIGN
```

The output from RandomForest Classifier on the dataset. Value of Percent Split data = 66%.



Classifier: RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options: Percentage split % 66

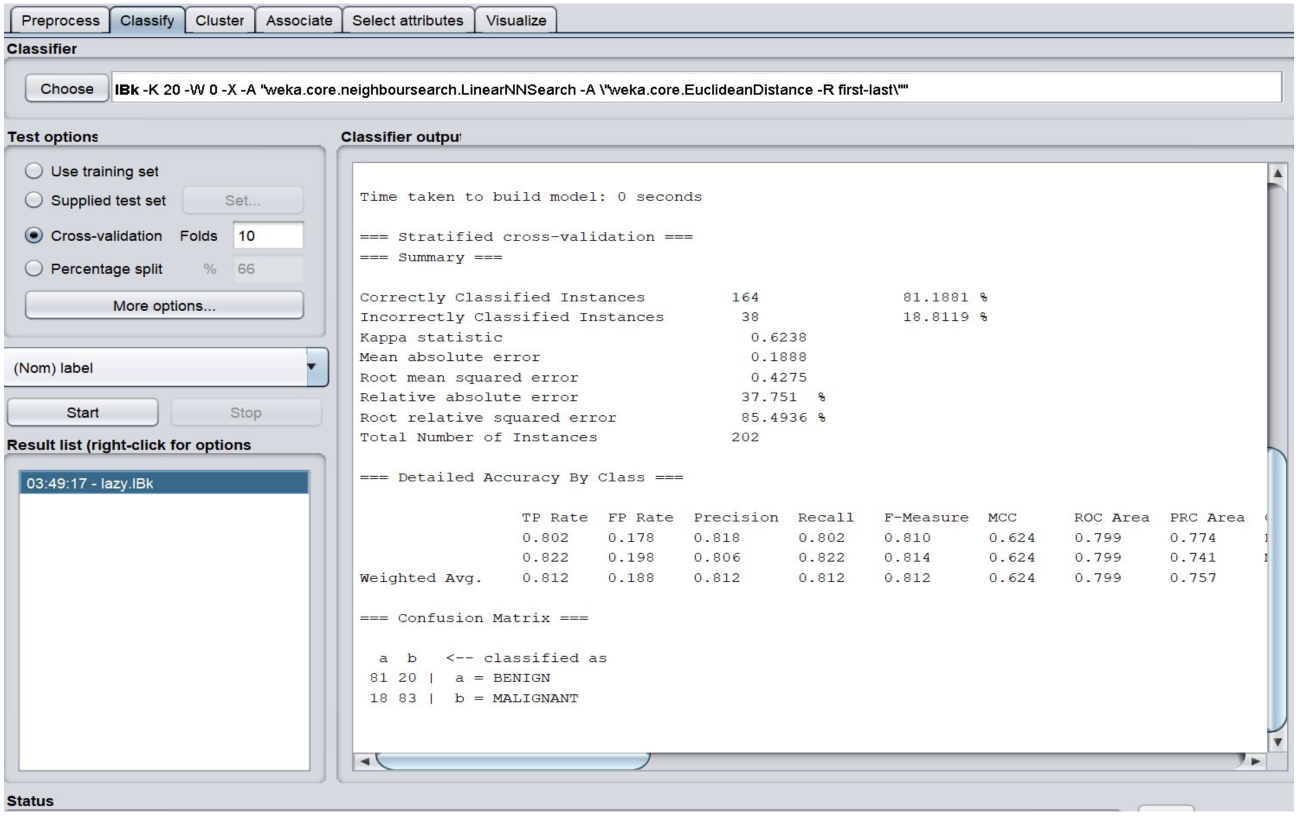
(Nom) label

Result list: 12:52:35 - trees.RandomForest; 12:54:13 - trees.RandomForest

```
=== Summary ===

Correctly Classified Instances          60               86.9565 %
Incorrectly Classified Instances         9               13.0435 %
Kappa statistic                          0.7433
Mean absolute error                      0.2362
Root mean squared error                  0.322
Relative absolute error                 46.9898 %
Root relative squared error             63.9733 %
Total Number of Instances               69

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.763    0.000    1.000      0.763    0.866      0.769    0.929     0.956     BENIGN
                 1.000    0.237    0.775      1.000    0.873      0.769    0.929     0.891     MALIGNANT
Weighted Avg.    0.870    0.106    0.899      0.870    0.869      0.769    0.929     0.927

=== Confusion Matrix ===

  a  b   <-- classified as
 29  9 |  a = BENIGN
  0 31 |  b = MALIGNANT
```

k-NN OUTPUTS

The output from IBk Classifier on the dataset. Number of folds in Cross-Validation = 10



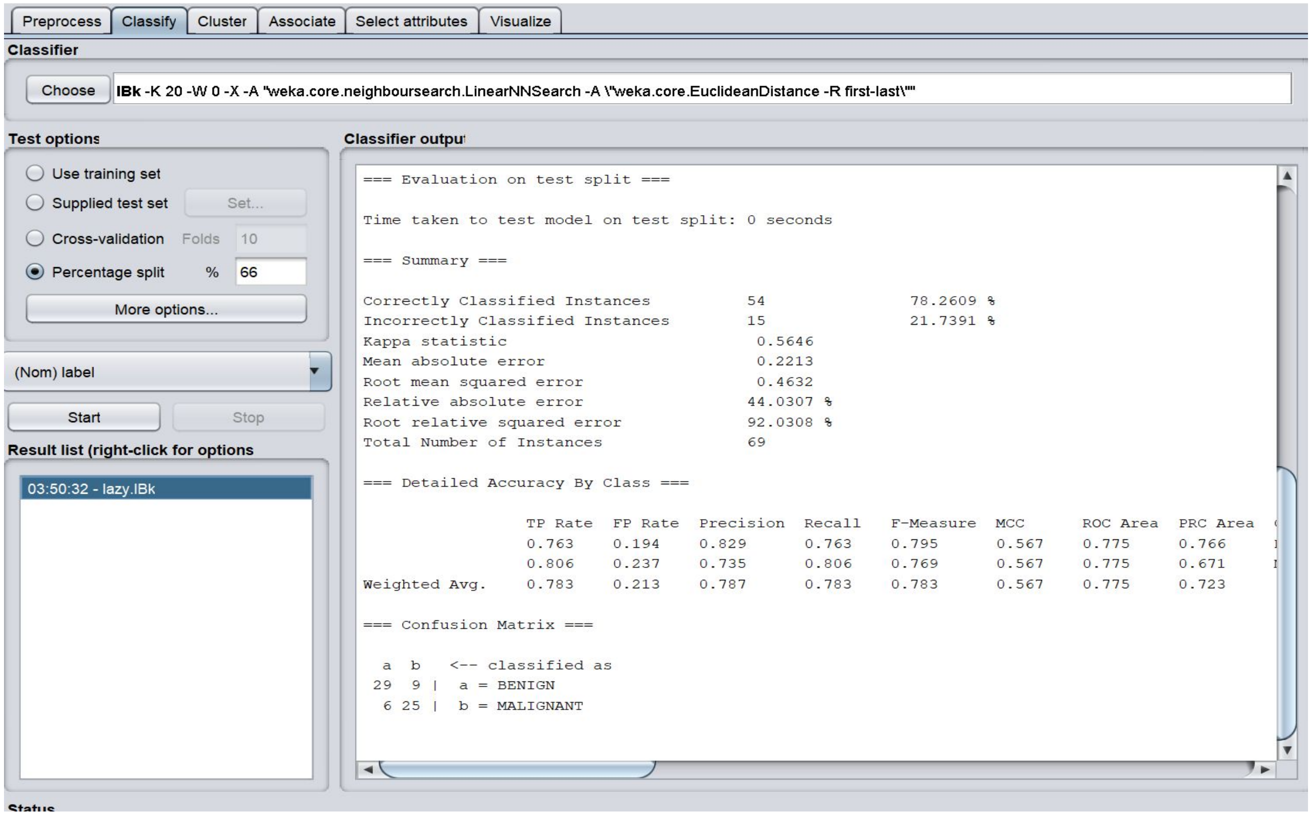
IBk -K 20 -W 0 -X -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Use training set
Supplied test set
[✓ Cross-validation] Folds 10
Percentage split % 66
(Nom) label

03:49:17 - lazy.IBk

```
Time taken to build model: 0 seconds

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

Correctly Classified Instances         164               81.1881 %
Incorrectly Classified Instances        38               18.8119 %
Kappa statistic                          0.6238
Mean absolute error                      0.1888
Root mean squared error                  0.4275
Relative absolute error                 37.751  %
Root relative squared error             85.4936 %
Total Number of Instances              202

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area
                 0.802    0.178    0.818      0.802    0.810      0.624    0.799     0.774
                 0.822    0.198    0.806      0.822    0.814      0.624    0.799     0.741
Weighted Avg.    0.812    0.188    0.812      0.812    0.812      0.624    0.799     0.757

=== Confusion Matrix ===

  a  b   <-- classified as
 81 20 |  a = BENIGN
 18 83 |  b = MALIGNANT
```

The output from IBk Classifier on the dataset. Value of Percent Split data = 66%.



IBk -K 20 -W 0 -X -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

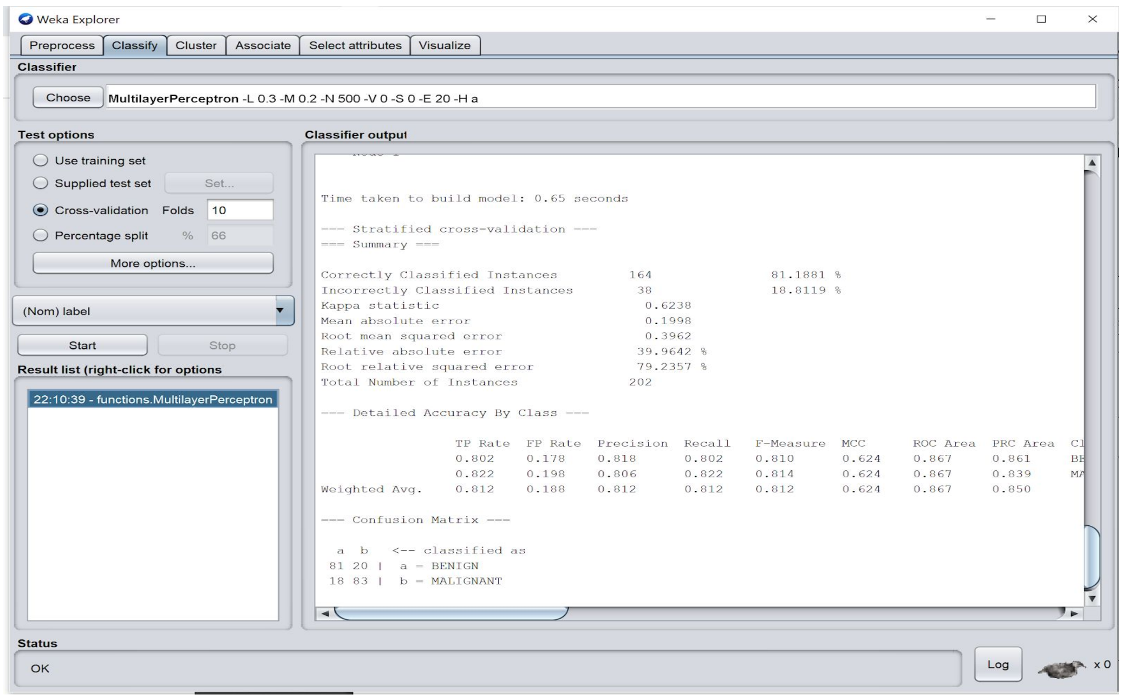
Use training set
Supplied test set
Cross-validation Folds 10
[✓ Percentage split] % 66
(Nom) label

03:50:32 - lazy.IBk

```
=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          54               78.2609 %
Incorrectly Classified Instances        15               21.7391 %
Kappa statistic                          0.5646
Mean absolute error                      0.2213
Root mean squared error                  0.4632
Relative absolute error                 44.0307 %
Root relative squared error             92.0308 %
Total Number of Instances               69

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area
                 0.763    0.194    0.829      0.763    0.795      0.567    0.775     0.766
                 0.806    0.237    0.735      0.806    0.769      0.567    0.775     0.671
Weighted Avg.    0.783    0.213    0.787      0.783    0.783      0.567    0.775     0.723

=== Confusion Matrix ===

  a  b   <-- classified as
 29  9 |  a = BENIGN
  6 25 |  b = MALIGNANT
```

ANN OUTPUTS

The output from MultilayerPerceptron Classifier on the dataset. Number of folds in Cross-Validation = 10



Classifier: MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a

Test options: Cross-validation Folds 10

(Nom) label

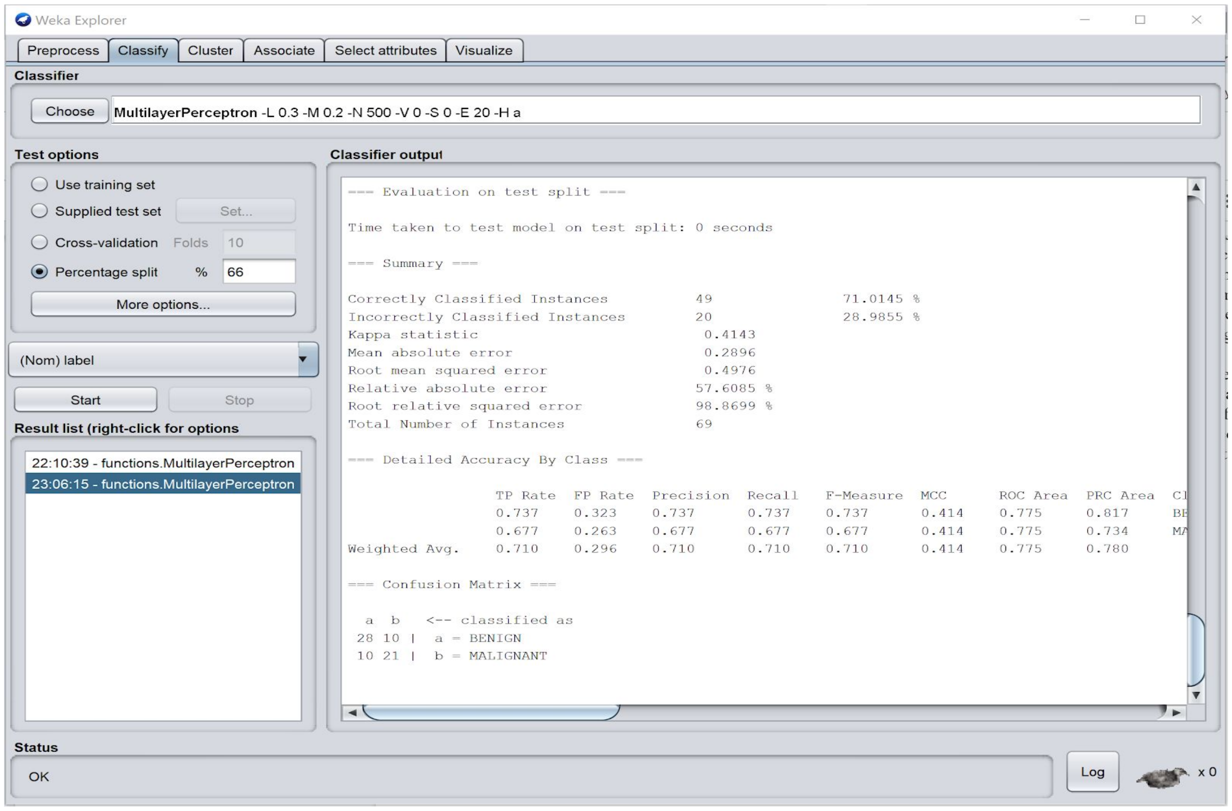
Result list (right-click for options): 22:10:39 - functions.MultilayerPerceptron

Classifier output

```
Time taken to build model: 0.65 seconds

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

Correctly Classified Instances         164               81.1881 %
Incorrectly Classified Instances        38               18.8119 %
Kappa statistic                          0.6238
Mean absolute error                      0.1998
Root mean squared error                  0.3962
Relative absolute error                 39.9642 %
Root relative squared error             79.2357 %
Total Number of Instances              202

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cl
                 0.802    0.178    0.818      0.802    0.810      0.624    0.867     0.861     BE
                 0.822    0.198    0.806      0.822    0.814      0.624    0.867     0.839     MA
Weighted Avg.    0.812    0.188    0.812      0.812    0.812      0.624    0.867     0.850

=== Confusion Matrix ===

  a  b   <-- classified as
 81 20 |  a = BENIGN
 18 83 |  b = MALIGNANT
```

Status: OK

The output from MultilayerPerceptron Classifier on the dataset. Value of Percent Split data = 66%.



Classifier: MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a

Test options: Percentage split % 66

(Nom) label

Result list (right-click for options): 22:10:39 - functions.MultilayerPerceptron; 23:06:15 - functions.MultilayerPerceptron

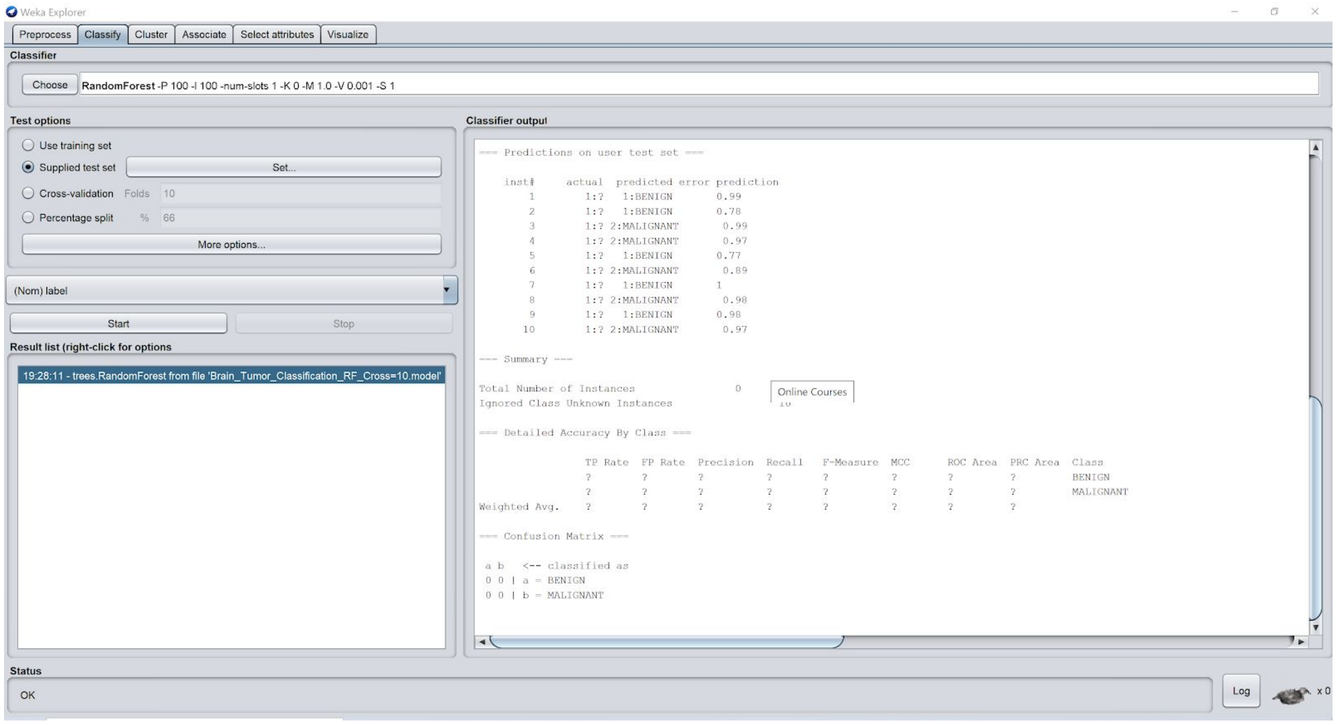
Classifier output

```
=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          49               71.0145 %
Incorrectly Classified Instances        20               28.9855 %
Kappa statistic                          0.4143
Mean absolute error                      0.2896
Root mean squared error                  0.4976
Relative absolute error                 57.6085 %
Root relative squared error             98.8699 %
Total Number of Instances               69

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cl
                 0.737    0.323    0.737      0.737    0.737      0.414    0.775     0.817     BE
                 0.677    0.263    0.677      0.677    0.677      0.414    0.775     0.734     MA
Weighted Avg.    0.710    0.296    0.710      0.710    0.710      0.414    0.775     0.780

=== Confusion Matrix ===

  a  b   <-- classified as
 28 10 |  a = BENIGN
 10 21 |  b = MALIGNANT
```

Status: OK

SUPPLIED TEST SET OUTPUTS

The output from the Supplied Test Dataset with model as Random Forest Classifier with Number of folds in Cross-Validation = 10 (as it produces better accuracy among % split and cross-validation)



Weka Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Classifier

Choose RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options

- Use training set
- [✓ Supplied test set] Set...
- Cross-validation Folds 10
- Percentage split % 66

(Nom) label

Result list (right-click for options)

19:28:11 - trees.RandomForest from file 'Brain_Tumor_Classification_RF_Cross=10.model'

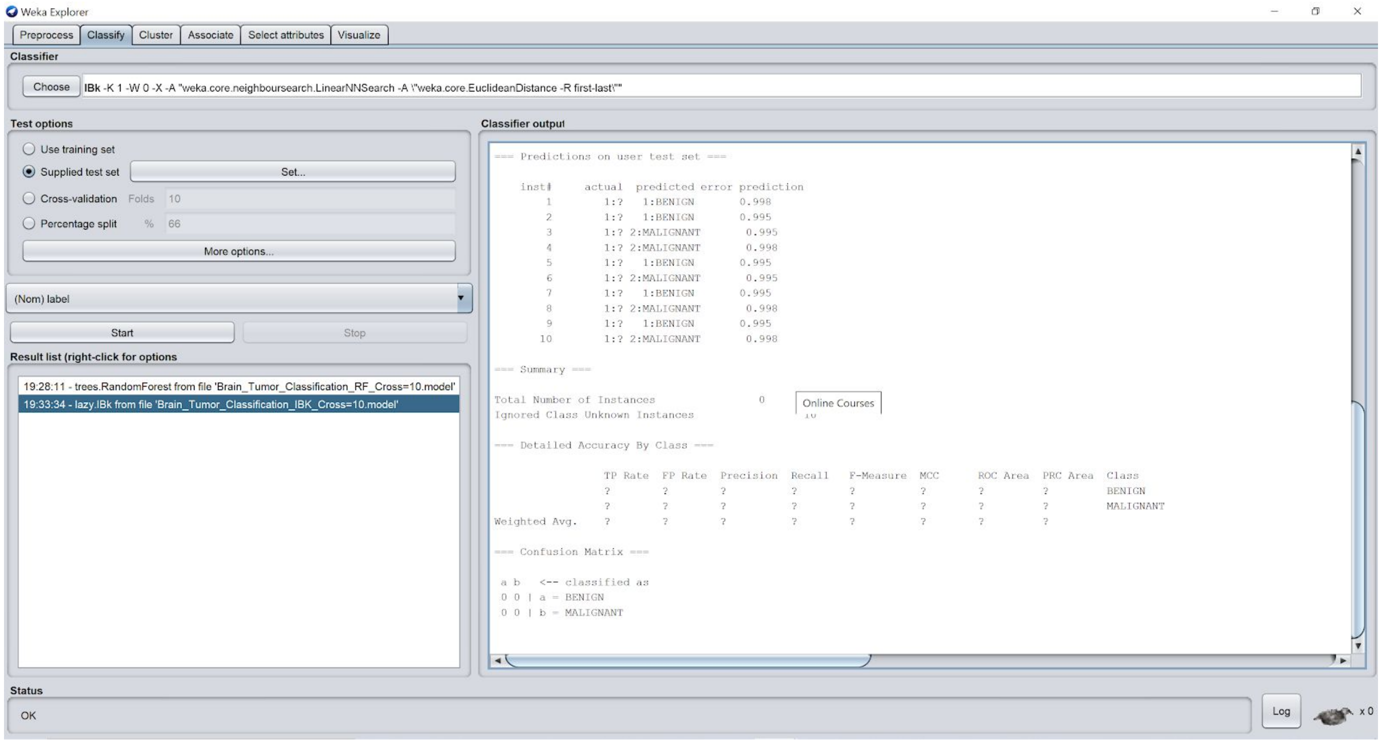
Classifier output

```
=== Predictions on user test set ===

    inst#     actual  predicted error prediction
        1        1:?   1:BENIGN       0.99
        2        1:?   1:BENIGN       0.78
        3        1:? 2:MALIGNANT       0.99
        4        1:? 2:MALIGNANT       0.97
        5        1:?   1:BENIGN       0.77
        6        1:? 2:MALIGNANT       0.89
        7        1:?   1:BENIGN       1
        8        1:? 2:MALIGNANT       0.98
        9        1:?   1:BENIGN       0.98
       10        1:? 2:MALIGNANT       0.97

=== Summary ===

Total Number of Instances                0
Ignored Class Unknown Instances                10

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 ?        ?        ?          ?        ?          ?        ?         ?         BENIGN
                 ?        ?        ?          ?        ?          ?        ?         ?         MALIGNANT
Weighted Avg.    ?        ?        ?          ?        ?          ?        ?         ?

=== Confusion Matrix ===

 a b   <-- classified as
 0 0 | a = BENIGN
 0 0 | b = MALIGNANT
```

Status

OK

The output from the Supplied Test Dataset with model as k-Nearest Neighbours Classifier with Number of folds in Cross-Validation = 10 (as it produces better accuracy among % split and cross-validation)



Weka Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Classifier

Choose IBk -K 1 -W 0 -X -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Test options

- Use training set
- [✓ Supplied test set] Set...
- Cross-validation Folds 10
- Percentage split % 66

(Nom) label

Result list (right-click for options)

19:28:11 - trees.RandomForest from file 'Brain_Tumor_Classification_RF_Cross=10.model'
19:33:34 - lazy.IBk from file 'Brain_Tumor_Classification_IBK_Cross=10.model'

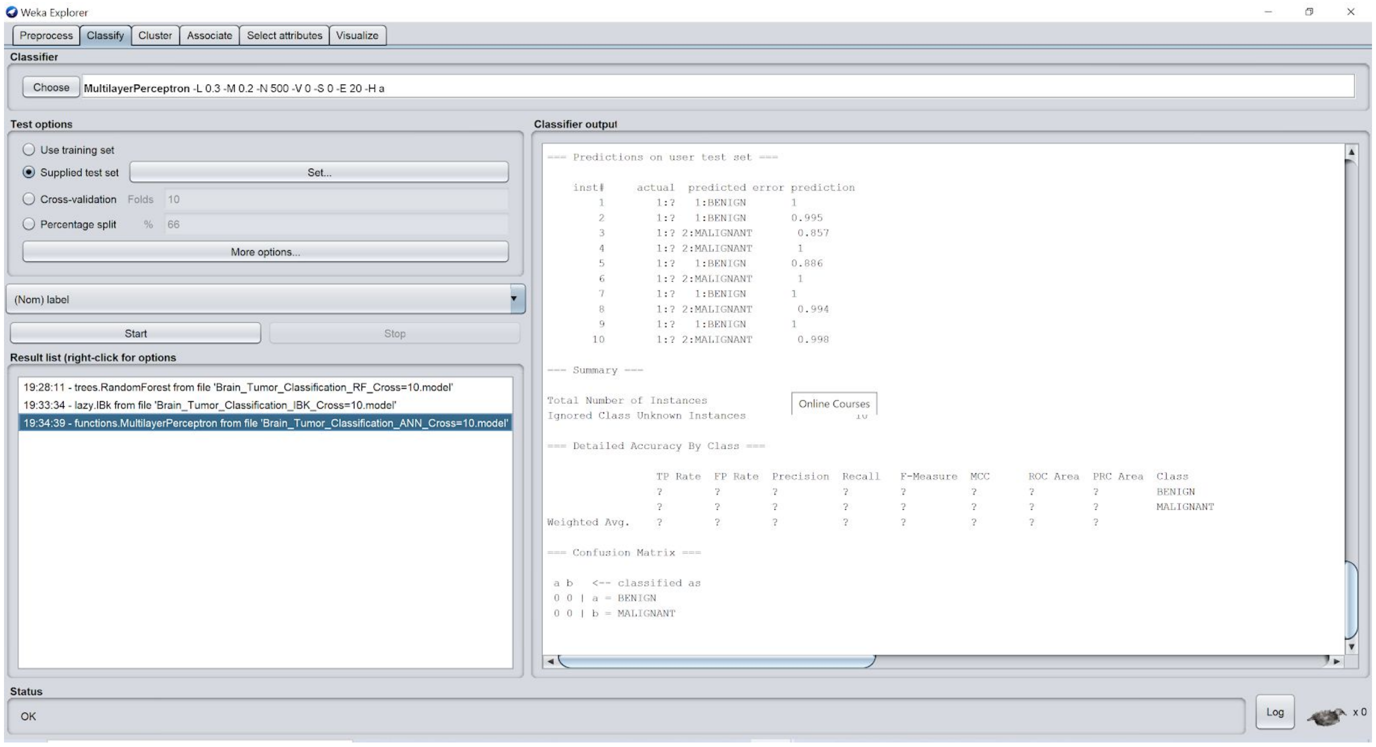
Classifier output

```
=== Predictions on user test set ===

    inst#     actual  predicted error prediction
        1        1:?   1:BENIGN       0.998
        2        1:?   1:BENIGN       0.995
        3        1:? 2:MALIGNANT       0.995
        4        1:? 2:MALIGNANT       0.998
        5        1:?   1:BENIGN       0.995
        6        1:? 2:MALIGNANT       0.995
        7        1:?   1:BENIGN       0.995
        8        1:? 2:MALIGNANT       0.998
        9        1:?   1:BENIGN       0.995
       10        1:? 2:MALIGNANT       0.998

=== Summary ===

Total Number of Instances                0
Ignored Class Unknown Instances                10

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 ?        ?        ?          ?        ?          ?        ?         ?         BENIGN
                 ?        ?        ?          ?        ?          ?        ?         ?         MALIGNANT
Weighted Avg.    ?        ?        ?          ?        ?          ?        ?         ?

=== Confusion Matrix ===

 a b   <-- classified as
 0 0 | a = BENIGN
 0 0 | b = MALIGNANT
```

Status

OK

The output from the Supplied Test Dataset with model as Artificial Neural Networks Classifier with Number of folds in Cross-Validation = 10 (as it produces better accuracy among % split and cross-validation)



Classifier: MultilayerPerceptron -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a

Test options: Use training set / [✓ Supplied test set] / Cross-validation Folds 10 / Percentage split % 66

(Nom) label

Result list (right-click for options)

```
19:28:11 - trees.RandomForest from file 'Brain_Tumor_Classification_RF_Cross=10.model'
19:33:34 - lazy.IBk from file 'Brain_Tumor_Classification_IBK_Cross=10.model'
19:34:39 - functions.MultilayerPerceptron from file 'Brain_Tumor_Classification_ANN_Cross=10.model'
```

Classifier output

```
=== Predictions on user test set ===

    inst#     actual  predicted error prediction
        1        1:?   1:BENIGN       1
        2        1:?   1:BENIGN       0.995
        3        1:? 2:MALIGNANT       0.857
        4        1:? 2:MALIGNANT       1
        5        1:?   1:BENIGN       0.886
        6        1:? 2:MALIGNANT       1
        7        1:?   1:BENIGN       1
        8        1:? 2:MALIGNANT       0.994
        9        1:?   1:BENIGN       1
       10        1:? 2:MALIGNANT       0.998

=== Summary ===

Total Number of Instances
Ignored Class Unknown Instances                 10

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 ?        ?        ?          ?        ?          ?        ?         ?         BENIGN
                 ?        ?        ?          ?        ?          ?        ?         ?         MALIGNANT
Weighted Avg.    ?        ?        ?          ?        ?          ?        ?         ?

=== Confusion Matrix ===

 a b   <-- classified as
 0 0 | a = BENIGN
 0 0 | b = MALIGNANT
```

Status: OK

Filenames and files -

Dataset Files -
Brain_Tumor_Classification_Train_Data.arff - Contains the Train Dataset after Removal of Outliers and Extreme Values.

Brain_Tumor_Classification_Supplied_Test.arff - Contains the Test Dataset.

Model Files stored in Model Folder -
Brain_Tumor_Classification_kNN_Cross=10.model - Contains model for IBk Classifier with Cross Validation = 10.
Brain_Tumor_Classification_kNN_%_Split=66%.model - Contains model for IBk Classifier with Percent Split = 66%.
Brain_Tumor_Classification_RF_Cross=10.model - Contains model for Random Forest Classifier with Cross Validation = 10.
Brain_Tumor_Classification_RF_%_Split=66%.model - Contains model for Random Forest Classifier with Percent Split = 66%.
Brain_Tumor_Classification_ANN_Cross=10.model - Contains model for Multilayer Perceptron Classifier with Cross Validation = 10.
Brain_Tumor_Classification_ANN_%_Split=66%.model - Contains model for Multilayer Classifier Perceptron with Percent Split = 66%.

7 REFERENCES -

[1]. **Brain Tumor Education** Retrieved from https://www.abta.org/.

[2]. Arora A et al(2018) Classification of brain tumor using devernay sub-pixel edge detection and k-nearest neighbours methodology. *Neuroimmunol Neuroinflammation* 2018;5:26. http://dx.doi.org/10.20517/2347-8659.2018.11

[3]. Guo, Gongde & Wang, Hui & Bell, David & Bi, Yaxin. (2004). KNN Model-Based Approach in Classification. Available at: *https://www.researchgate.net/publication/2948052*

[4]. Ali, Jehad & Khan, Rehanullah & Ahmad, Nasir & Maqsood, Imran. (2012). Random Forests and Decision Trees. *International Journal of Computer Science Issues(IJCSI)*. 9.

[5]. Kustrin, Snezana & Beresford, Rosemary. (2000). Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis*. 22. 717-27. 10.1016/S0731-7085(99)00272-1.

[6]. **Weka wiki** Retrieved from https://waikato.github.io/weka-wiki/documentation/

[7]. Powers, David & Ailab,. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. **J. Mach. Learn. Technol.** 2. 2229-3981.10.9735/2229-3981.