G53MLE Assignment 2: Decision Tree

Date: 17th Nov 2016

Group 3 Members:
Zisi Ye
Yiming Zhang
Zhihan Zhang
Afraz Hussain
Zher Shin Tan

1. Introduction

The second assignment asks the group to create and train six binary classification trees by using provided facial point data. The group named six trees: Tree 1, Tree 2, Tree 3, Tree 4, Tree 5, and Tree 6. Each one of these trees corresponds to one emotion. For example, if the group tests one set of facial point data in the Tree 6, and the outcome is 1, then this means this set of facial point data is the emotion 'surprise'.

Following is a brief overview of how the team achieved this task:

Firstly, the group processed the input data. In the file 'label2bin.m', it transferred one set of data y to six sets of same size data, however the new data only contains 0 and 1.

Next, the group followed ID3 algorithm pseudo code below to build the decision tree.

```
ID3 (Examples, Target_Attribute, Attributes)
    Create a root node for the tree
    If all examples are positive, Return the single-node tree Root, with label = +.
    If all examples are negative, Return the single-node tree Root, with label = -.
    If number of predicting attributes is empty, then Return the single node tree Root,
    with label = most common value of the target attribute in the examples.
    Otherwise Begin
        A ← The Attribute that best classifies examples.
        Decision Tree attribute for Root = A.
        For each possible value, vi, of A,
            Add a new tree branch below Root, corresponding to the test A = vi.
            Let Examples(vi) be the subset of examples that have the value vi for A
            If Examples(vi) is empty
                Then below this new branch add a leaf node with label = most common target value in the examples
            Else below this new branch add the subtree ID3 (Examples(vi), Target_Attribute, Attributes – {A})
    End
    Return Root
```

'CreateDecisionTree.m' , 'bestFeature.m', 'entropy.m', 'gain.m' implemented the ID3 algorithm.

Finally, the group used 10 fold cross validation to train and test the decision tree. Also, the confusion matrix has been plotted and hence calculated the recall, precision, F1 measure. ‘cw2.m’, ‘crossvalidation.m’, ‘DrawConfusionMatrix.m’, ‘dtclassify.m’, ‘Fmeausre.m’, ‘evaluation.m’ are used in the evaluation part.

2. Run the code

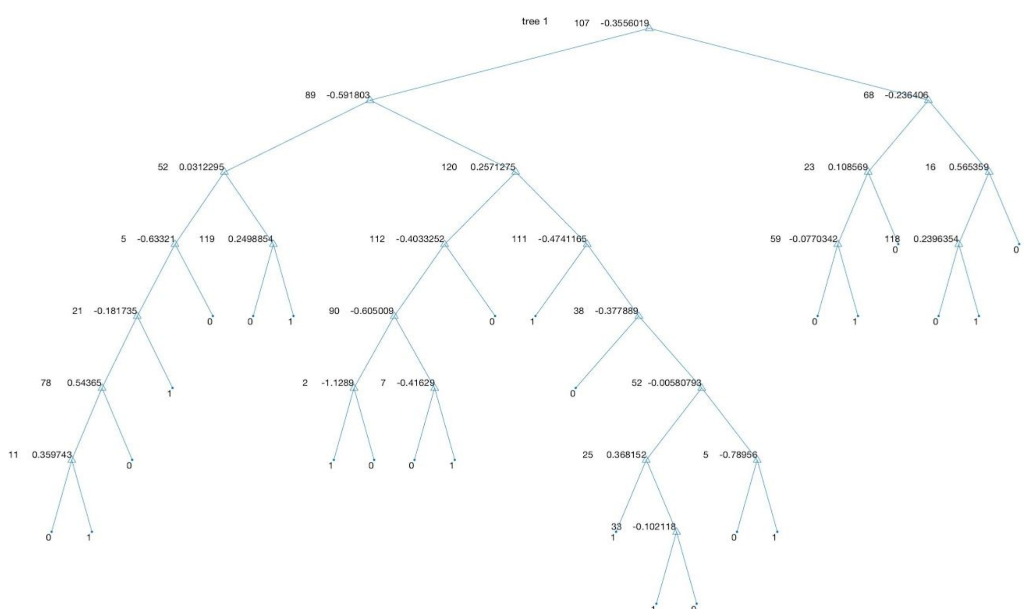
It may require workspace to be cleared first.
‘cw2_1.m’ : plots 6 trees.
‘cw2.m’: results are 10 matrices(cross validation) and a result matrix named ‘result’, also can plot 60 trees.
‘evaluation.m’: after run this file, created recall, precision and F1 measure for the ‘result’ matrix.
The rest files are all functions that are needed in these three files.

3. Results

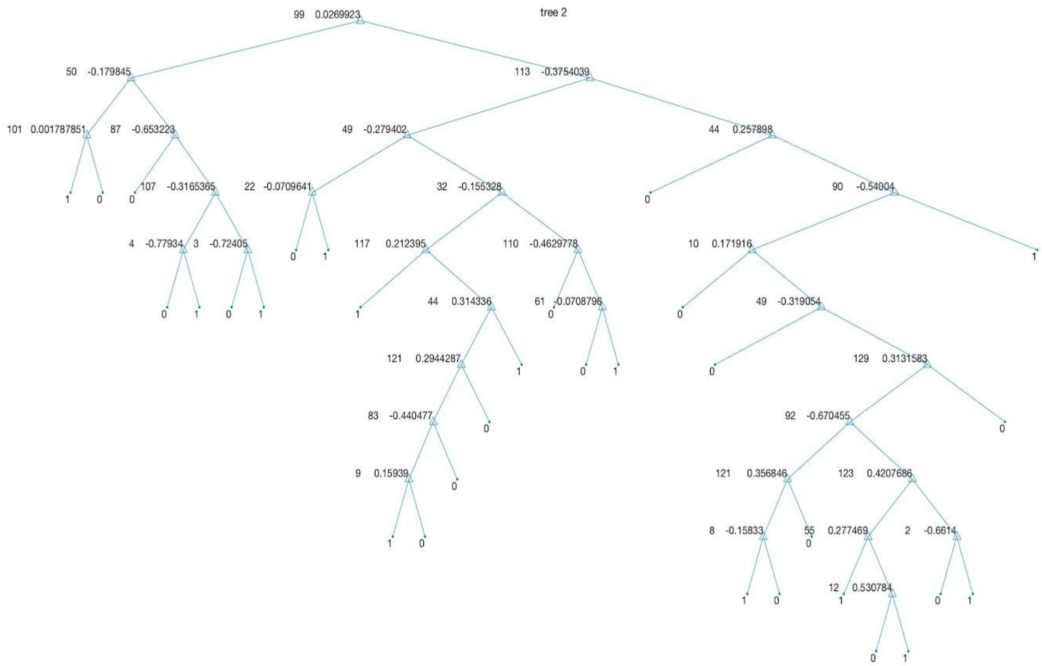
<u>3.1 Decision Trees graphs</u>

These 6 trees below are trained and plotted by using 612 sets of facial data(x). The group wrote the script ‘cw2_1.m’ to implement this. Each node has two values (binary), the first value represents the best attribute, and the second value is the best threshold. Attributes are basically named from 1 to 132 just as what it is given in the data x.

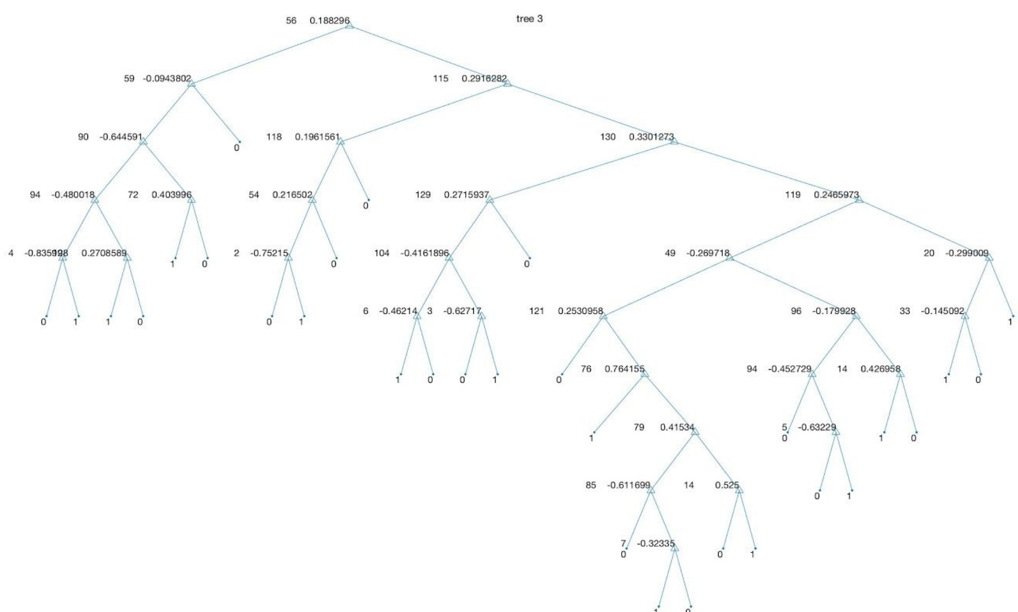
Tree 1(target label is 1)



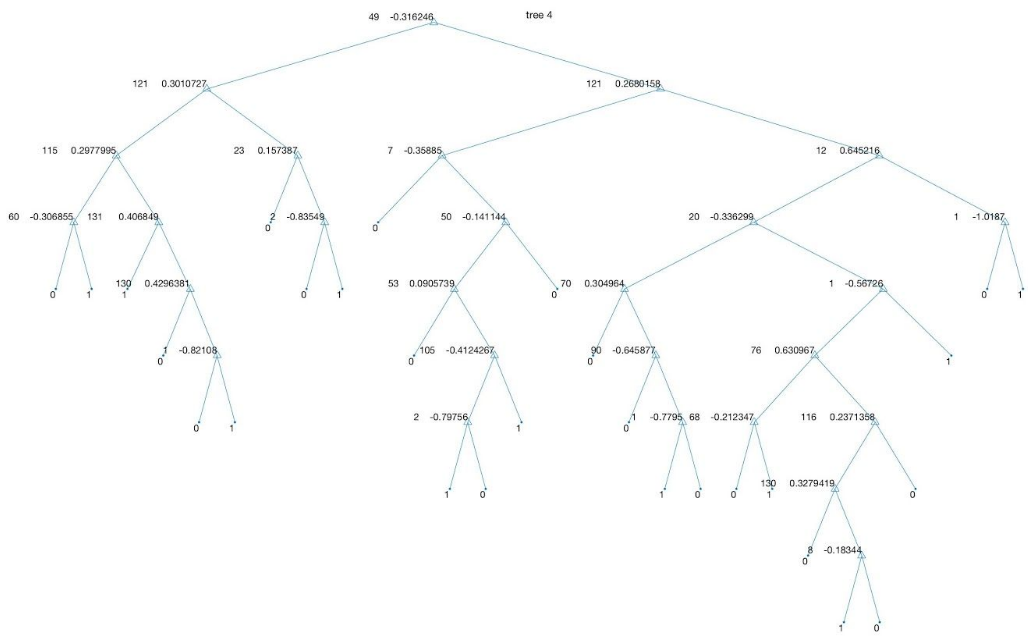
Tree 2(target label is 2)



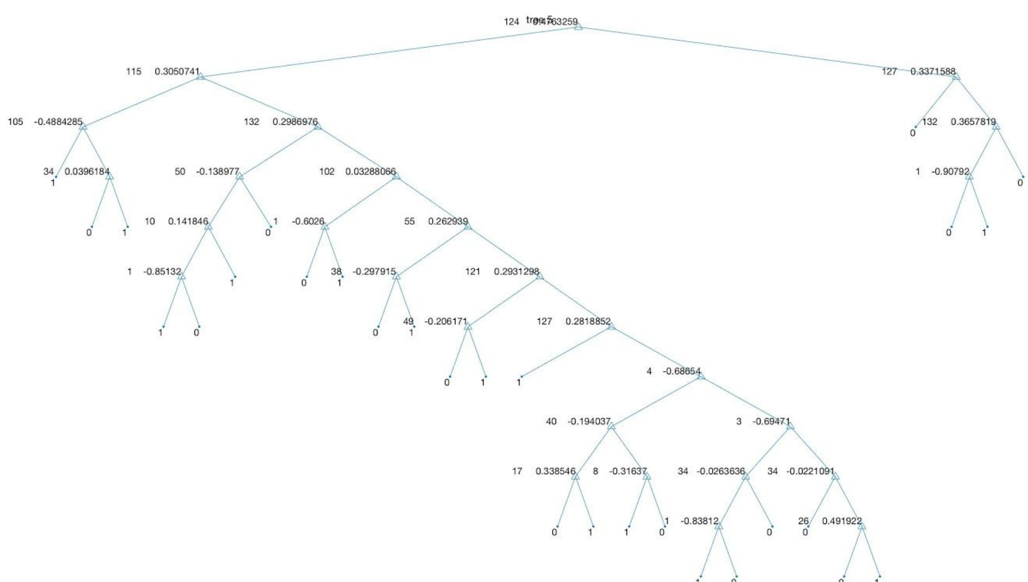
Tree 3(target label is 3)



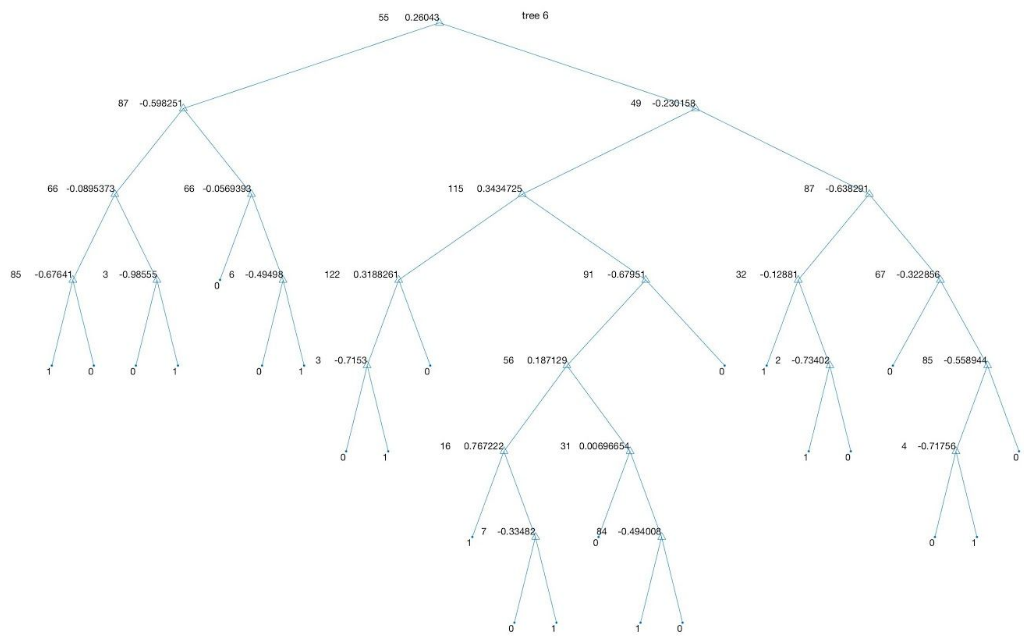
Tree 4(target label is 4)



Tree 5(target label is 5)



Tree 6 (target label is 6)



<u>3.2 Confusion matrix</u>

This part result was run by ‘cw2.m’. The group used 10 fold cross validation, this means the group split original 612 sets of data into 10 folds. Then 1 fold was used to test and rest 9 folds were used to train the decision tree. After each fold has already been used to test, the group got 10 confusion matrix. Notice that the group deleted the input data when they are failed to classify into any of the emotion. Then, the group summed 10 confusion matrices to get the result average confusion matrix. False negatives are cases predicated negative which are actual positive. So in this question, what the group deleted can be seen as false negatives. In confusion matrix below , the sum of all elements is 537, namely, 75 sets of input data are false negatives.

confusion matrix

	Target 1	Target 2	Target 3	Target 4	Target 5	Target 6
Output 1	34	22	2	3	7	6
Output 2	9	31	13	9	7	5
Output 3	6	2	34	13	8	12
Output 4	1	1	5	95	2	7
Output 5	4	6	9	3	39	8
Output 6	1	2	3	3	2	78

3.3 Average precision, recall rates and F_1-measure

The group wrote the file 'evaluation.m' to calculate the recall, precision and F1 measure. The calculation of them are based on the function:
Recall = TP/TP + FN
Precision = TP/ TP + FP
F1 measure = 2*(Precision*Recall)/(Precision + Recall)
By evaluating them, the group can find out the decision trees' behaviour. From the chart, Tree 2 was trained poorest and Tree 4 was the best training tree.

average recall, precision and F1 measure

	Target 1	Target 2	Target 3	Target 4	Target 5	Target 6
recall	61.82%	48.44%	51.52%	75.40%	60.00%	67.24%
precision	45.95%	41.89%	45.33%	85.59%	56.52%	87.64%
F1 measure	52.71%	44.93%	48.23%	80.17%	58.21%	76.10%

<u>3.4 Difference between the original tree and pruned tree</u>

Pruning is a frequently used technique which helps to reduce error, cost complexity and also prevent overfitting of a decision tree. However, choosing the right node to prune is a hard decision as the selected node might not reduce the error. The idea of applying pruning on a decision tree is to start pruning from leaf nodes to the root nodes. Image 1 shows the tree without pruning, and image 2 shows the tree after pruning. (This tree was actually Tree 1 in the part 3.1.)

Consider the tree shown in Image 1(each node has been assigned with two numbers which indicates the number of positive examples and negative examples respectively.): there are 8 pairs of leaf nodes in the tree. To determine which pair of nodes is to be pruned first, the impurity penalty is calculated by getting the opposite value of gain from each node. Then the nodes with lowest gain is selected to be pruned first. This is because the gain of separating the nodes into two leaves is lower, the impurity penalty of merging these leaves is lower than others as well. As it is shown in Image 1, the nodes in red circle is determined to be pruned first. After being pruned, the tree becomes what is shown in Image 2. The node ‘1 22’ becomes a leaf node and has class ‘0’ as the majority of the examples have label ‘0’.

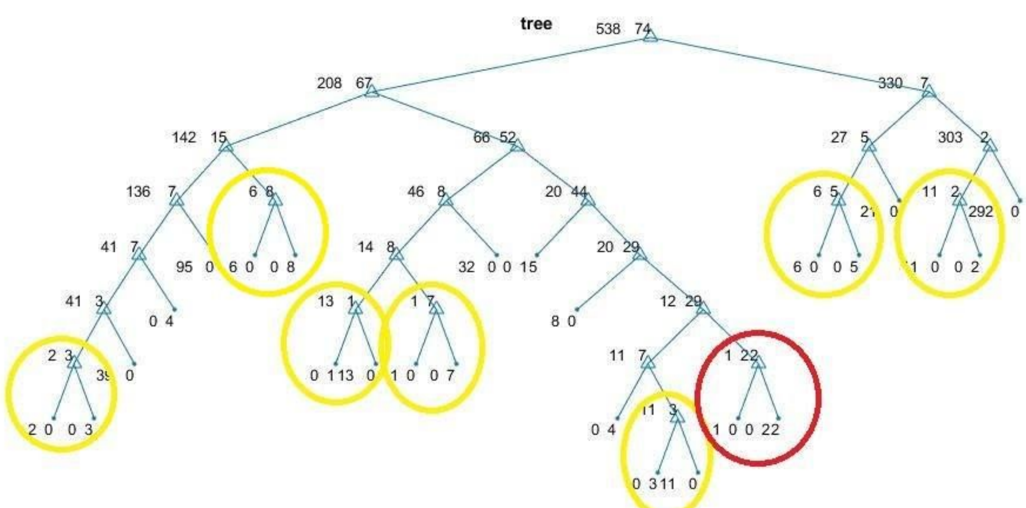


Image 1: Tree before pruning.

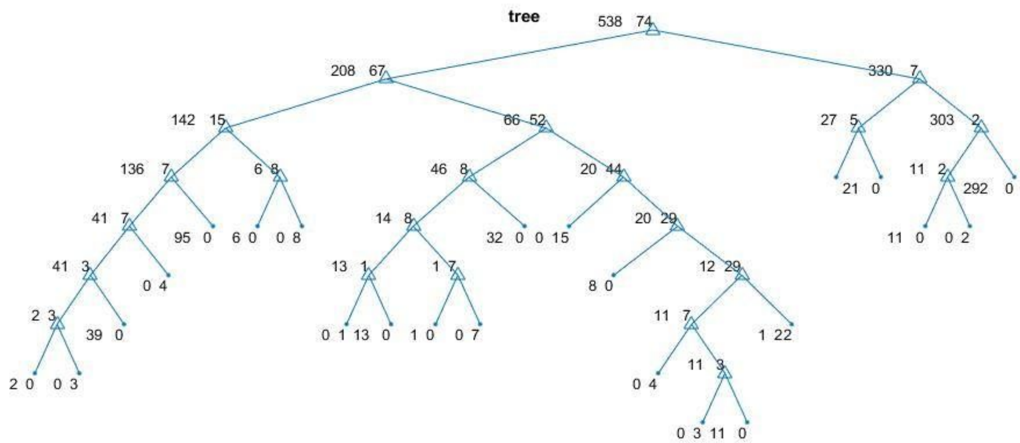


Image 2: Tree after pruning.

3.5 Explanation on final decision on what emotion is present

Given the condition that the team had six trained trees for the six different emotions. Then a test set which consists of 6 different emotions is tested on each tree. Each data of the test set has 132 attributes to represents an emotion, and those attributes are used to compare with the trained tree value. For instance, the root node of the tree1 which represents emotion 1 has a threshold of -0.3556 at attributes 107, so the data will be categorised based the value in attribute 107 whether it has smaller or larger number than the calculated threshold. This process is repeated until the data has reached the leaf node which has only either 1 or 0. If the tree returns 1 to the data, that means the data is categorised as emotion 1. For the data which categorised as 0 will further proceed to tree2, tree3 and so on until the data is classified.

However, when the group tested the tree using 10 fold cross validation, the outcome showed that most sets of test data will result into only one emotion, but some sets of test data will not match any emotion, or some will match more than one emotion. To solve this problem, for the multiple result case, the group determined to use the first emotion as it found when processed from Tree 1 to Tree 6. Also, the group deleted the related results when no emotion can be found. Because if the output is all 0, then it does not exist true positives and false positives. The sum of the elements of the final confusion matrix will smaller than 612.

3.6 Ways to have a single tree that could perform classification directly

If the group would like to change the code in order that a single tree can classify 6 emotions, then firstly, the group doesn't need to process the input data y, it can just be numbers from 1 to 6 where each number corresponds to one emotion. Next, the tree is no longer a binary classification tree, because if the group build the decision tree, the leaf nodes should be ranged from 1 to 6 instead of only 1 and 0. This change can be done in the file ‘ CreateDecisionTree’ by implementing new base cases for 1, 2, 3, 4, 5, 6 to decide how to reach a leaf node in this case. For example , the pseudocode to implement leaf node of value 6 can be like: ‘ If all examples are 6 then return the single-node tree root with label = 6.’ However, the difficulty of this problem is to determine when does all the nodes are 6. This should be carefully considered and maybe write other functions to help achieve that.

4. Conclusion

The group implemented ‘id3 algorithm’ on the matlab and evaluated the trained decision trees by using cross validation and confusion matrix. Based on the evaluation result, for some emotions, the F1 measure rate seems good, however for other emotions, the decision tree doesn’t classify well.