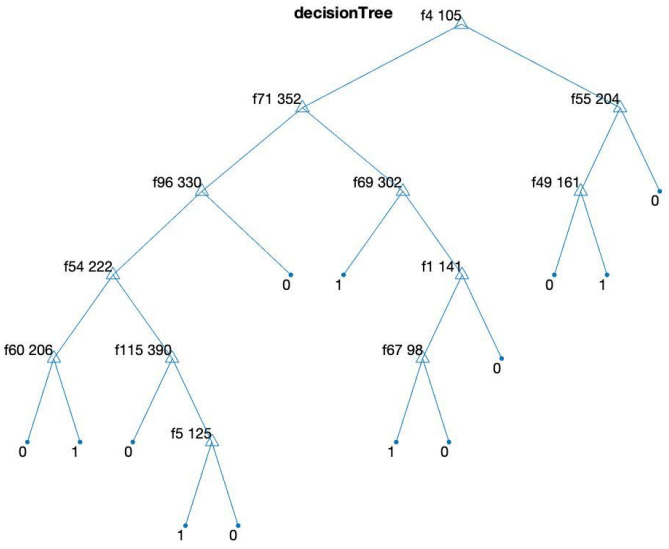
Assignment 2 Report

Part 1: Results

The acquired decision tree.



The above tree is generated by using the whole data set as training set. At each internal node, the node's name is combined by two parts, the best feature and the best threshold.

Cross validation classification results:

- Recall and precision rates.

Trail No	Precision Rate	Recall Rate
1	1.00	0.83
2	0.71	0.83
3	1.00	0.80
4	0.83	0.83
5	1.00	0.83
6	1.00	0.83
7	1.00	0.80
8	0.71	0.83
9	0.80	0.80
10	0.67	0.80

- The F1-measure derived from the recall and precision rates of the previous step.

Trail No	F1 Score
1	0.91
2	0.77
3	0.89
4	0.83
5	0.91
6	0.91
7	0.89
8	0.77
9	0.80
10	0.73

Part 2: Questions

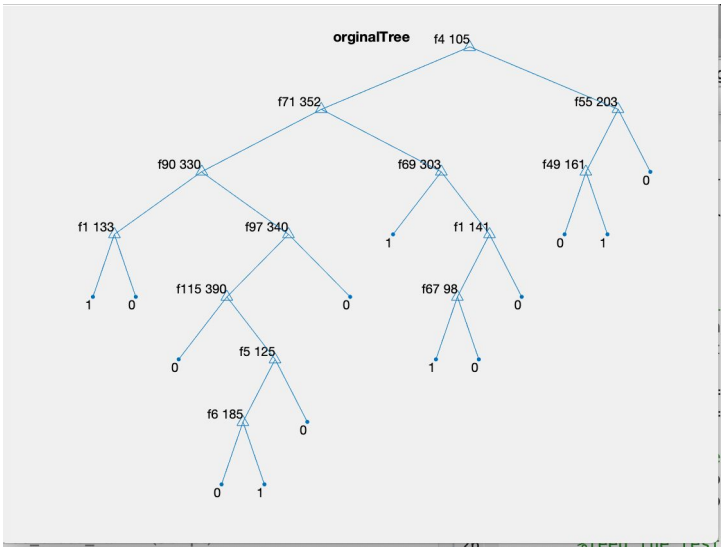
Pruning is an important issue in trees. Explain what pruning does:

Pruning is a technique that reduces the size of decision trees by removing subtrees that provide little power to classify instances. It reduces the complexity of the final classifier, and hence improves predictive accuracy by reduction of overfitting.

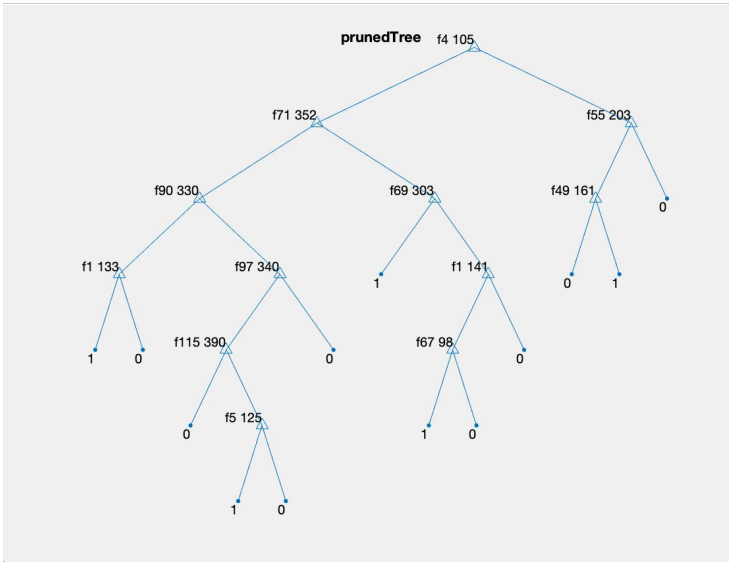
There are two way of performing the pruning process: pre-pruning and post-pruning.
Pre-pruning removes the nodes during construction of the tree (and only the training set is used) by setting constraints. Several types of constraint can be used, such as a maximum depth of the tree, the minimum amount of examples in a leaf and a minimum gain info for a split.
Post-pruning constructs a complete decision tree, allowing the tree to overfit the training data, then replacing those subtrees (internal nodes) with insufficient confidence with leaf nodes whose class labels are marked with the most frequent classes in the node subtree. The main post-pruning algorithms are Reduced-Error Pruning(REP), Pessimistic-Error Pruning (PEP) and Error-Complexity Pruning(ECP). This method is preferred over pre-pruning because it is difficult to estimate when to stop tree growth in the pre-pruning method.

Find the node(s) that would be pruned first for one of your learned trees.

Firstly, we used post-pruning with REP for pruning our learned decision tree. The principle of REP is if the F1 score of the new tree would be equal to or greater than that of the original tree and that subtree contains no subtree with the same property, then the subtree is replaced by a leaf node. REP starts from the bottom-up. For example, a decision tree (trained in cross validation) shown below.



The first node that would be considered to be pruned is ‘f6 185’. The f1 score of the original decision tree F(t) = 0.80. If the node is pruned, then the F1 score become F’(t) = 0.833. Therefore, the node ‘f6 185’ would be pruned first in this decision tree. And the pruned tree would look like:



Another way to do pruning is called Error Complexity Pruning, finds the error complexity for each node and prunes the node with the minimum error complexity. The error cost is calculated by:

$R(t) = r(t) * p(t)$, where $r(t) = \frac{no\ of\ examples\ missclassified\ in\ node}{no\ of\ all\ examples\ in\ node}$ and $p(t) = \frac{no\ of\ examples\ in\ node}{no\ of\ total\ examples}$

If node t was not pruned then error cost of subtree T, rooted at t:

$$R(T) = \sum_{i = node\ of\ leaves} R(i)$$

The error complexity of the node is given as:

$$a(t) = \frac{R(t) - R(T)}{no\ of\ leaves - 1}$$

For the original tree shown above, by calculating a(t) for each node, we can find a(t) for node 'f6 185' is the minimum a(t):

$$a(t) = \frac{1/15 - 2/15}{2 - 1} = -\frac{1}{15}$$

Therefore, the node 'f6 185' also will be pruned first by using this method.

Explain the difference between the original and the pruned tree.

As shown above, the pruned tree has a smaller size and depth is reduced by 1. The pruned node is replaced by a leaf node where the class is assigned by the majority value of its parent node. The prediction accuracy and generalization ability of pruned tree is improved by reducing its complexity and reducing overfitting.

In this assignment, you trained a binary tree. Explain how you would use decision trees to learn to predict the multiclass problem of six-basic emotion recognition. Explain how to make decisions in leaf nodes.

Decision trees for the multiclass assignment problem operate much in the same way as those designed for binary classification. The attribute and threshold that provide the best split at each node drive its expansion, sending the examples that test positive on said threshold to one of two newly-created branches, and the rest to the remaining one. Assuming there is no premature stopping criterion, the tree will stop its expansion when a potential split would produce a node with no examples, or all the examples in a resulting node belong to a single class. The second of these cases is the simplest one, and the corresponding node is simply declared to be a leaf whose label takes the value referencing this class. The first one, on the other hand, is resolved again by making the node a leaf, where a majority vote on class is cast among the examples it contains to determine its label. Once the tree has been created, its usage is the standard one for decision trees. For a new test example, its attributes are systematically tested against the thresholds set at each node, for which afterwards the corresponding branch (positive or negative) is chosen until a leaf node is reached. At this point, the test example is said to "belong" to the class referenced by the leaf node label.
Another way of designing decision trees for the multiclass problem is by considering a higher branching factor for the tree, meaning that it ceases to be binary as multiple splits are

performed at each node. This is typically the case when attributes having discrete, unordered values (a.k.a nominal data), so it becomes more convenient to think of categories instead of thresholds as separation criteria.

And how you would have to change your query search algorithm for any node?

According to the literature (Duda and Hart, Chapter 8), the use of a gini impurity measure (replacing function I(·,·) in the Lab Manual) is suggested as a metric to calculate the optimal attribute/threshold combination for the splitting of the nodes in the case of multiclass classification:

$$i(N) = \frac{1}{2}\left[1 - \sum_j P^2(\omega_j)\right]$$

Where ω_j is the proportion of examples in the node belonging to class *j*.
Additionally, for the calculation of the maximum information gain to be obtained at each split, the Remainder(·) function has to be modified so that it considers the proportions of elements belonging to all classes; that is, the parameter *v* in the summation must be increased from 2 to 6:

$$Rem(A) = \sum_{i=1}^{v} \frac{pi+ni}{p+n} I\left(\frac{pi}{pi+ni}, \frac{ni}{pi+ni}\right)$$