Assignment 2: Decision Trees Algorithm
— Report —

Robert Jaworzyn, Aodhgan Gleeson, Ben Fadero, Levi Epstein.
{rj616, amg315, bof16, lbe16 }@doc.ic.ac.uk

Teaching helper: Jiankang Deng
Course: CO395, Imperial College London

10th February, 2017

1 Implementation Details

1.1 Selecting the best attribute in each node

The ID3 algorithm is a decision tree learning algorithm which selects attributes for nodes based on their information gain. The attribute with the highest information gain at any point is the best candidate for a node.

This algorithm generally works with positive and negative examples for a given target. However, the data in question deals with 6 different targets (emotions, in this case). It is therefore necessary to train 6 separate trees, counting each emotion as a binary target - the decision tree for happiness, for example, would treat happy examples as positive and all other examples as negative.

To implement this concept in MATLAB, a function called chooseBestDecisionAttribute was created. This function was applied to each emotion. The function selects the attribute with the highest Information Gain in the following way:

- The number of positive and negative examples are counted and the total entropy is calculated using these values
- The function then iterates through each attribute, calculating its information gain. The information gain is the reduction in total entropy caused by partitioning the set of data according to the attribute in question (possibly reference lect notes).
- The function stores the highest information gain calculated across all attributes, and this attribute is selected as the best attribute for the node.

1.2 Performing cross-validation

In general, cross-validation is performed by splitting the data into k folds, using k-1 folds for training and validation, and the final fold for testing. This process is then repeated k times, using a different fold as the testing fold each time.

For the purposes of this task, the data was divided into 10 roughly evenly-sized folds (9 for training and 1 for testing). Using the training data, a tree was trained for each of the 6 emotions, and the perfomance of each tree was tested by comparing its predictions against the data provided. This was implemented by doing the following:

- A row of attributes from the testing data was passed through the trained trees (1 to 6), thus giving a prediction of which emotion the row of attributes corresponds to. This process was repeated for all rows in the testing data.
- The predictions were then compared against the actual data by creating a confusion matrix (see blah).

1.3 Computing the average results

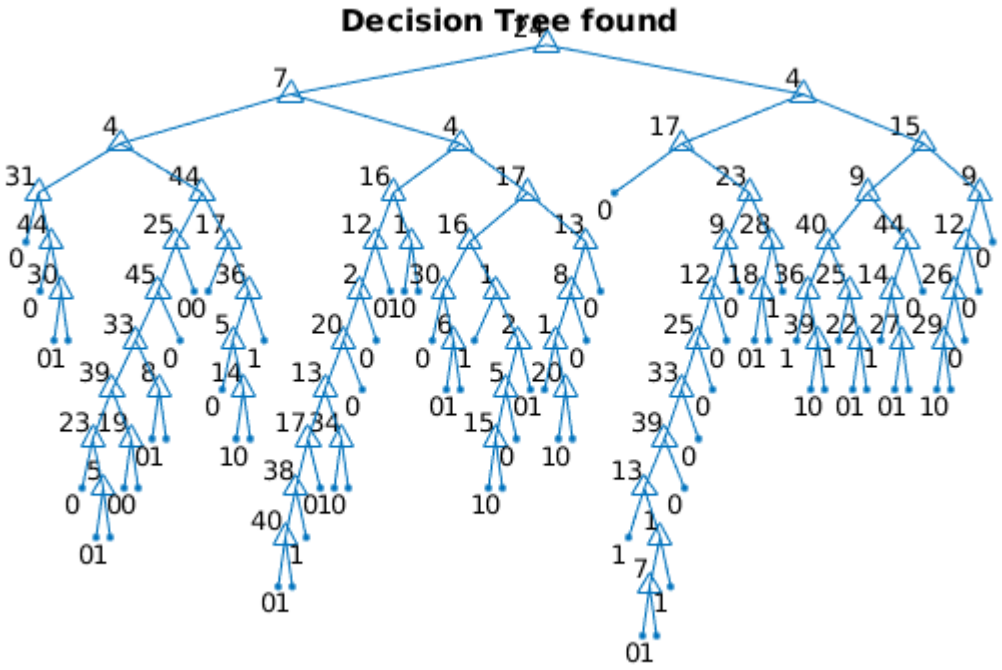
A confusion matrix was created for each of the iterations of the 10-fold cross-validation. The ten confusion matrices were then summed together to give a cumulative confusion matrix, from which the average precision, recall, F1 and classification rate could be calculated using the appropriate formulae.

1.4 Other system implementation details

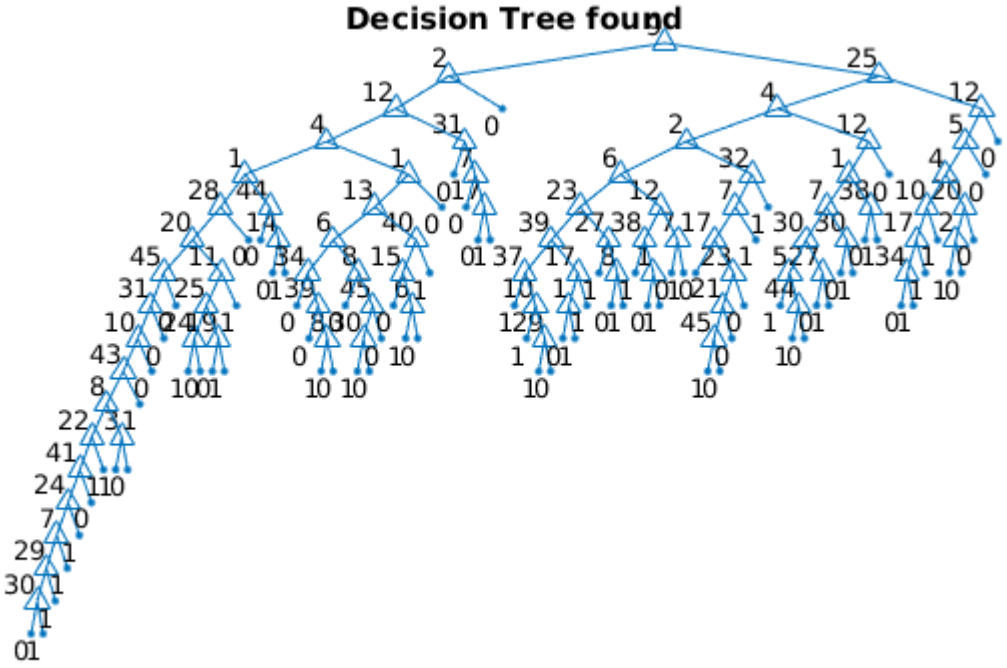
This is where we should put a bit about using the random evaulation method ... (see section 4.2)

2 Tree Figures

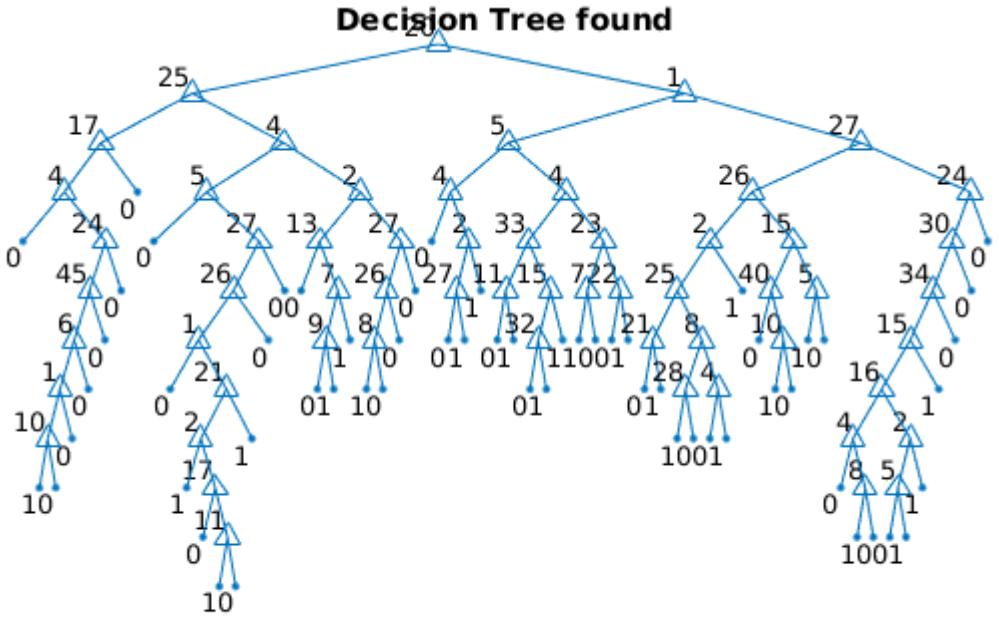
2.1 Emotion 1



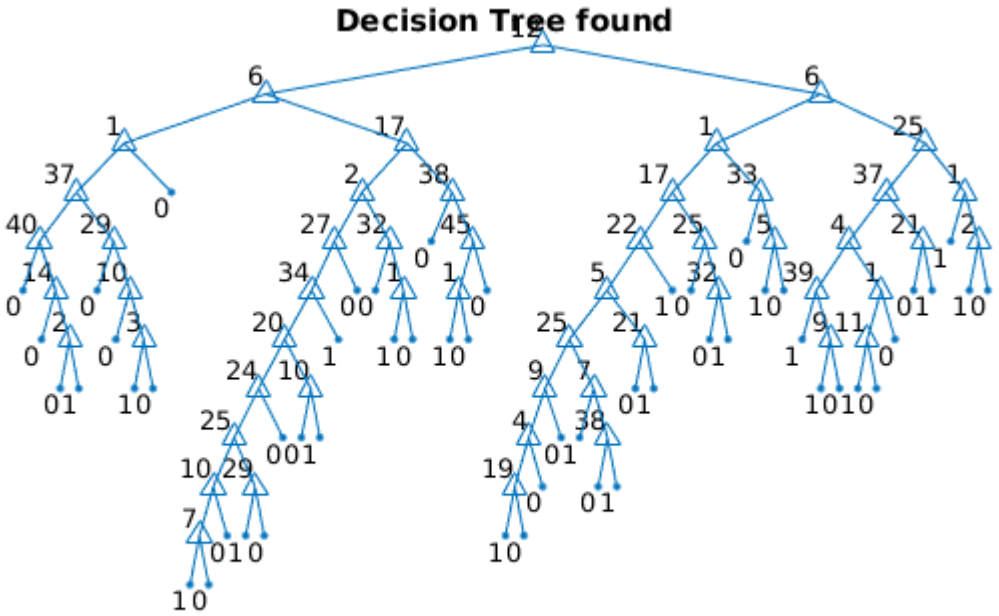
2.2 Emotion 2



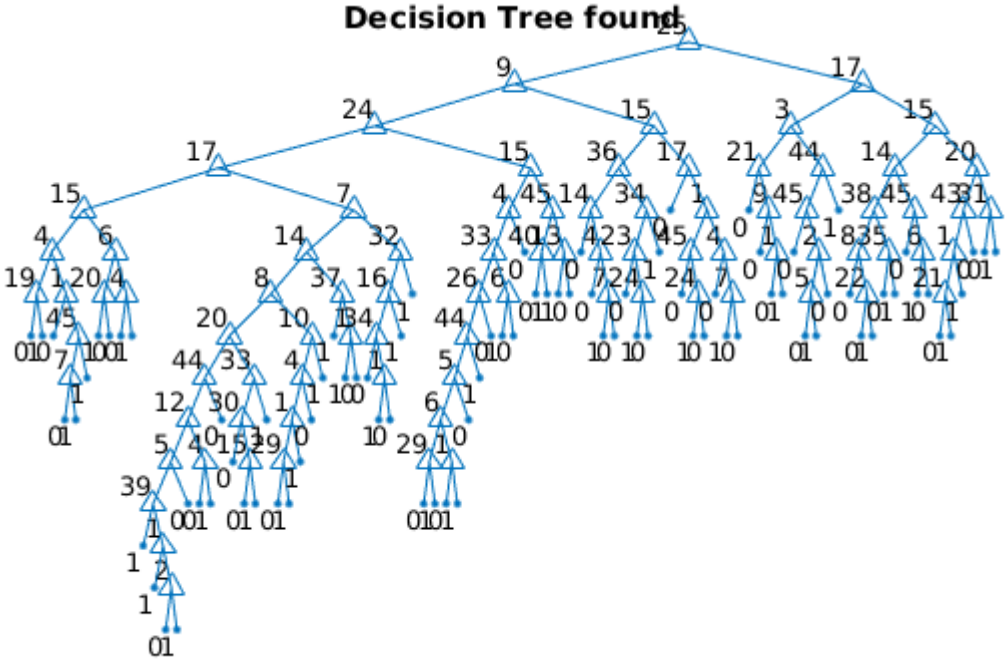
2.3 Emotion 3



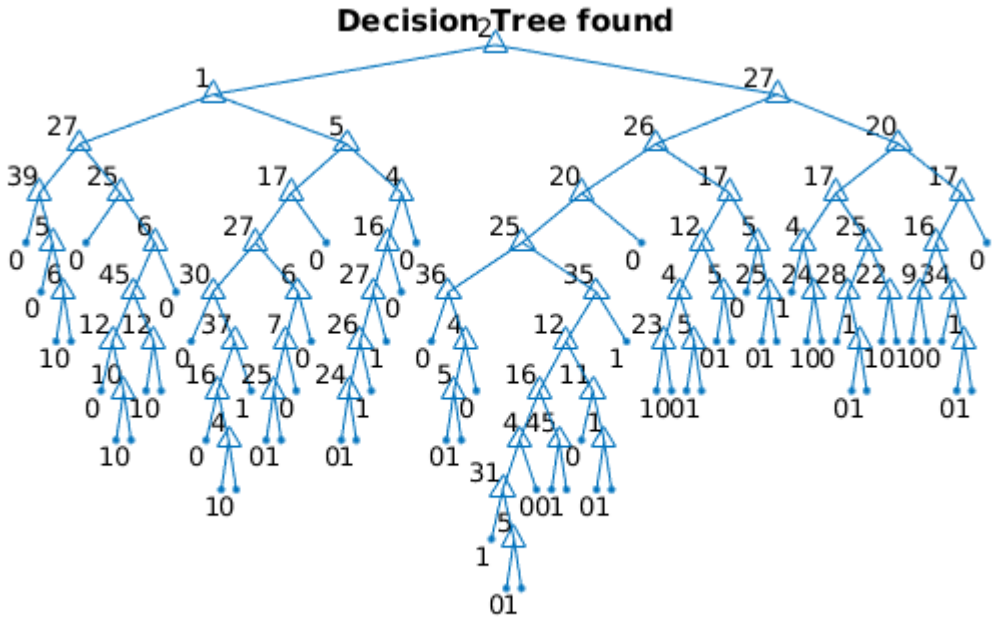
2.4 Emotion 4



2.5 Emotion 5



2.6 Emotion 6



3 Results of the Evaluation

Including the average confusion matrix, the average classification rate and the average precision, recall rates and F1-measure for each of the 6 classes; for both clean and noisy datasets

3.1 For the clean data set

3.1.1 The average confusion matrix

		Predicted Class					
		1	2	3	4	5	6
Actual Class	1	84	18	9	3	14	4
	2	19	146	6	9	10	8
	3	7	4	77	5	12	14
	4	2	12	5	180	12	5
	5	23	17	8	7	68	9
	6	5	6	13	8	10	165

The classes 1-6 represent the 6 different emotions (anger, disgust, fear, happiness, sadness and surprise). Looking at the emotion 1 class as an example, the confusion matrix shows that of the 132 actual examples of emotion 1, the system predicted 84 correctly as emotion 1 (anger), 18 as emotion 2 (disgust), 9 as emotion 3 (fear), 3 as emotion 4 (happiness), 14 as emotion 5 (sadness) and 6 as emotion 6 (surprise). This means that there were 84 True Positives (TPs) and 48 False Negatives (FNs) for the emotion 1 class. There were 146, 77, 180 , 68 and 165 TPs for emotion 2,3,4,5,6 classes respectively.

3.1.2 Classification Rate

The average classification rate = 71.71%. This measures the number of correctly predicted examples divided by the total number of examples.

3.1.3 Precision Rate

The average precision rates for each of the 6 classes (in percentages):

Class					
1	2	3	4	5	6
60.00	71.92	65.25	84.91	53.97	80.49

3.1.4 Recall Rate

The average recall rates for each of the 6 classes (in percentages):

Class					
1	2	3	4	5	6
63.64	73.74	64.71	83.33	51.52	79.71

3.1.5 F1-measure

The F1-measure for each of the 6 classes (in percentages):

Class					
1	2	3	4	5	6
61.76	72.82	64.98	84.11	52.71	80.10

3.2 For the noisy data set

3.2.1 The average confusion matrix

	Predicted Class						
Actual Class		1	2	3	4	5	6
	1	21	14	15	13	15	10
	2	11	125	15	21	5	10
	3	16	20	94	22	14	21
	4	5	14	15	155	8	12
	5	19	9	14	6	49	13
	6	13	11	20	11	13	152

3.2.2 Classification Rate

The average classification rate = 59.54%

3.2.3 Precision Rate

The average precision rates for each of the 6 classes (in percentages):

Class					
1	2	3	4	5	6
24.71	64.77	54.34	67.98	47.12	69.72

3.2.4 Recall Rate

The average recall rates for each of the 6 classes (in percentages):

Class					
1	2	3	4	5	6
23.86	66.84	52.07	74.16	44.55	69.09

3.2.5 F1-measure

The F1-measure for each of the 6 classes (in percentages):

Class					
1	2	3	4	5	6
24.28	65.79	52.22	70.94	45.79	69.41

4 Questions

4.1 Noisy-Clean Datasets Question

There is a difference in performance when training testing using a clean data set in comparison with a noisy data set. When looking at the classification performance measures above (confusion matrix, classification rate, recall and precision rates, F1-measure) for the clean and noisy data sets, we can see that the performance of the classifiers when using the clean data set is superior to the performance when using the noisy data set.

- The average classification rate is 12.17% higher using clean data set.
- The precision rates using the clean data set are higher for each of the six classes.
- The recall rates using the clean data set are higher for each of the six classes.
- The F1-measures using the clean data set are higher for each of the six classes.

The clean data is considered to be completely accurate, whereas the noisy data contains some errors and incorrect information. The noisy data set used here could contain some incorrectly detected AUs and some AUs may be missing (reference CBC).

Since there may be some inccorrectly detected or missing AUs in the data, this would impact the performance of the learning algorithm. If the system is learning from data that contains several errors, it could increase the possibilty of the making incorrect predictions. This could lead to a smaller amount of True Positives and larger amount of False Negatives classified for each class. This is why the performance measures are worse when using the noisy data data set.

In essence, less accuracy in the data used for training, will result in less accuracy in classification of examples and this is evident in the results.

4.2 Ambuiguity Question

4.3 Pruning Question

How does the pruning example function work?

When a model being used to classify examples is too complex, it can lead to overfitting. Pruning is a method of removing subtrees and branches from a decision tree that do not aid in classifying examples. This makes the model less complex which helps to deal with the problem of overfitting.

classregtree constructs unpruned classification tree with two branches from each node , test function checks how ¡good¿ the tree is using two methods, resubstituation and cross validation.

Then prune tree with each test method

-red line is resubstitution- tree is trained and tested on the same data (if not pruned (at 200 nodes), is perfect because testing and training data is exactly the same - overfitted though!!). will always get worse as pruned more.

-blue line is 10-fold cross-validation - explains why cost is higher - trained and tested on different data. will have optimum tree size.