G53MLE - Decision Tree

Group 2 (Ting Pan, Junbin Wang, Yu Ding,Bolaji Abiodun,Bert Dzambulatov)

November 18, 2016

Introduction

This report details how the recognition of human facial expressions was achieved by using Decision Tree with ID3 Algorithm. To evaluate the performance and generalization of the decision trees, we conducted 10-cross validation and calculated precision, recall rates and F1-measure from generated confusion matrix. For each cross-iteration, 90% of samples are used for training and the rest 10% are feed for validation. Additionally, Details of ID3 algorithm implementation and its results are well explained in this report. Visualization graphs of the decision tree and matrix are attached for a clear demonstration of the results. Besides, the importance of pruning and how we combined 6 trees to a single prediction of the emotion are discussed. Finally, a way of producing a single tree that directly performs the multi-class classification is presented in the last part of the report.

1 Decision Tree Results

Following images demonstrate six trees generated from the dataset. Each image has a tag showing it's represented label. For each non-leaf node, there are two numbers. The first number is the attribute and the second number is the threshold of how to split the data. The leaf node is the binary classification result, which is either 1 or 0.

Figure 1: Graph of the decision tree for each emotion

2 Evaluation of the Decision Tree

The below two tables show the evaluation of the decision tree after 10-cross validation. Confusion matrix for every iteration of cross validation are included in appendix B.

	Label 1	Label 2	Label 3	Label 4	Label 5	Label 6
Label 1	25	15	5	9	15	5
Label 2	9	40	6	11	12	6
Label 3	2	11	39	10	14	14
Label 4	8	12	12	99	6	6
Label 5	16	7	5	6	42	8
Label 6	1	10	7	10	5	104

Table 1: Average Confusion Matrix after 10-cross validation

	Recall	Precision	F1 measure
Label 1	0.3378	0.4098	0.3704
Label 2	0.4762	0.4211	0.4469
Label 3	0.4333	0.5270	0.4756
Label 4	0.6923	0.6828	0.6875
Label 5	0.5000	0.4468	0.4719
Label 6	0.7591	0.7273	0.7429
Average	0.5331	0.5358	0.5325

Table 2: Evaluation for average and each label

3 Single Prediction with Six Trees

Six binary decision trees are generated for six emotion respectively. Each decision tree is capable of deciding if the sample belongs to a particular emotion. There are three different situations when combine the result of each tree and output a single decision.

- Only one tree has a positive output. In this case, the class of this tree is the classification result.
- Multiple trees have positive outputs. In this case, a tree will be randomly selected among them as the classification result of the sample.
- All tree have negative output. Since in this case, the sample does not belong to any label according to six decision trees, we randomly choose a label from all labels as the result of classification.

4 Pruning

In general, the decision tree algorithm will keep splitting the data and ends up with pure subsets. It enable the tree to perform extremely well in training dataset but may perform terribly in validation dataset. This problem is known as overfitting. Pruning is the measure to avoid the this problem. There is two type of pruning algorithms: Pre-Pruning and Post-Pruning. For Post-Pruning algorithms, Reduced-Error Pruning (REP) is simple and effective. The general idea of REP algorithm is that:

- Randomly divide data into training dataset and validation dataset.
- Training a decision tree based on all training dataset
- For each node on the current level of the tree:
  - Deleted the node and all children node of it, then measure the performance of the new tree after pruning on validation dataset.
  - If the performance is improved after pruning the node, then actually delete this node from the tree, otherwise keep it as before.
  - Repeat this process for one step up of the tree.

By applying REP algorithm to the original tree, the nodes which lower or not significantly contributed to the performance of the validation data will be pruned to reduce the tree complexity.

Compared with the original tree, the pruned tree is less likely to over-fitting and will have a simpler structure.

5 Multi-class Decision Tree

The decision tree can be applied to multi-class classification problems. One popular approach is Random Forest Algorithm, which takes random attributes to train multiple trees and predict the results based on the classification of all trees.

A single multi-class decision tree can be achieved by changing the formula for calculating the Entropy of a binary set to multi-class set. Given a set X contains n class, $P(x_i)$ is the proportion of the element with a certain class i in set X, the Entropy $H(X)$ of this set can be calculated by

$$H(x) = \sum_{i=1}^{n} -P(x_i) \log_2 P(x_i) \tag{1}$$

Applying this formula in code, we successfully generated a decision tree which can classify 6 emotions. Details can be refer to appendix A.

Appendix

A Multi-class Decisions Tree

For each non-leaf node, there are two numbers. The first number is the feature to be classified and the second number is the threshold used to determine how to split data. The leaf node is the classification result with only one number with value [1 , 6] indicating what how the data should be classified.

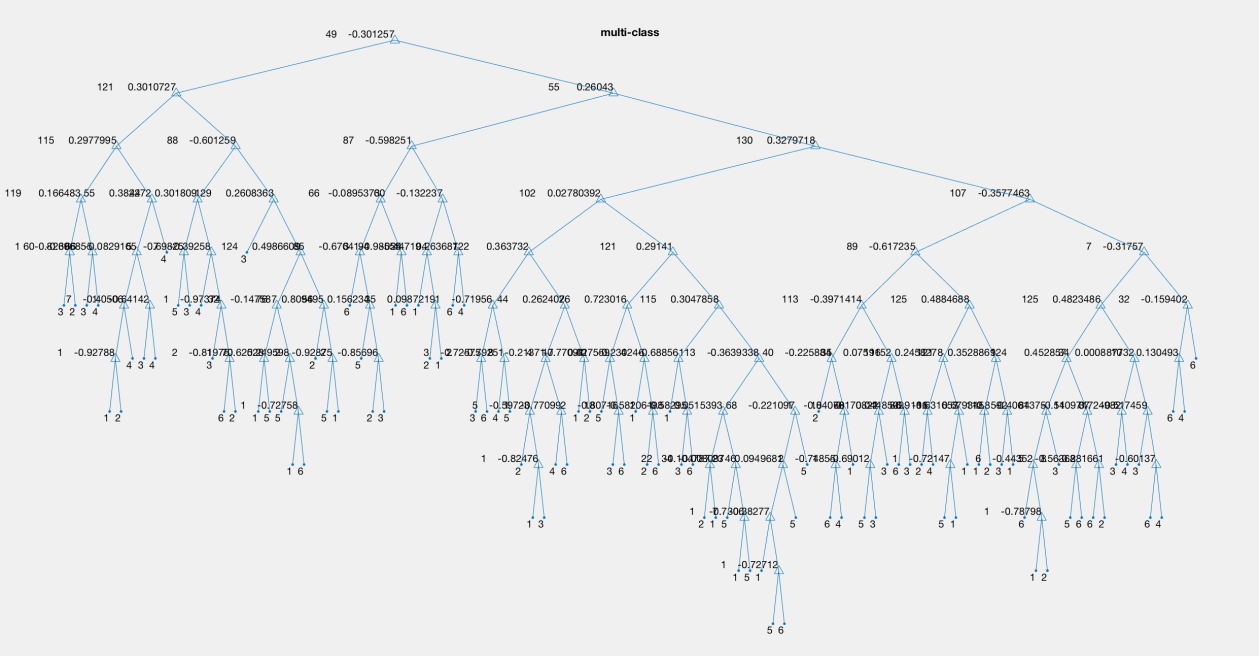

Figure 2: Graph for Multi-class Decision Tree

B Confusion Matrix for Each Iteration

Valiation_confusion_matrix(:,:,N) indicates the Nth confusion matrix. Every confusion matrix is 6 X 6 in which valiation_confusion_matrix(x,y,N) indicate that the number of samples that are with x as its actually value and y as the classification result of the decision tree.

validation_confusion_matrix(:,:,1) =

2	2	1	0	2	0
2	4	0	0	1	1
0	1	3	2	2	1
1	1	2	9	1	0
3	0	0	1	1	3
1	0	0	0	1	12

validation_confusion_matrix(:,:,2) =

4	1	0	0	1	2
0	4	2	2	1	0
1	0	6	0	1	1

```
     1     2     2     9     0     1
     1     1     0     1     6     0
     0     0     0     2     0    11
```

validation_confusion_matrix(:,:,3) =

```
     1     3     0     0     2     1
     0     5     2     1     0     0
     0     1     4     0     3     1
     0     0     2    11     0     1
     2     1     0     1     4     0
     0     2     0     2     2     8
```

validation_confusion_matrix(:,:,4) =

```
     4     0     0     2     1     1
     0     5     1     0     2     1
     0     0     6     1     0     2
     0     1     2    11     0     0
     2     1     1     0     5     0
     0     0     0     0     1    13
```

validation_confusion_matrix(:,:,5) =

```
     4     0     0     2     1     0
     1     4     0     1     2     0
     0     1     3     0     2     3
     1     1     0    12     1     0
     2     0     0     1     4     1
     0     1     0     2     0    11
```

validation_confusion_matrix(:,:,6) =

```
     1     2     1     2     1     0
     1     4     0     1     2     0
     1     2     1     0     3     2
     1     2     1     8     1     1
     3     1     1     1     2     0
     0     1     2     0     0    10
```

validation_confusion_matrix(:,:,7) =

```
     1     2     1     0     4     0
     3     2     1     1     1     1
     0     0     5     2     0     2
     2     0     2     7     2     1
     0     1     0     0     7     1
```

```
     0     2     2     2     0     8
```

validation_confusion_matrix(:,:,8) =

```
     2     2     1     0     2     0
     0     4     0     2     0     2
     0     1     2     3     2     1
     0     4     0     9     0     1
     1     1     1     0     5     0
     0     1     1     0     1    11
```

validation_confusion_matrix(:,:,9) =

```
     4     2     0     1     1     0
     1     3     0     1     3     1
     0     4     3     1     0     1
     1     1     0    13     0     0
     1     1     1     0     4     2
     0     0     2     0     0    11
```

validation_confusion_matrix(:,:,10) =

```
     2     1     1     2     0     1
     1     5     0     2     0     0
     0     1     6     1     1     0
     1     0     1    10     1     1
     1     0     1     1     4     1
     0     3     0     2     0     9
```