395 Machine Learning
CBC 1
Decision Trees

Ruikun Cao(rc2917), Yuxiang Wu(yw1117), Jiangbo Yu(jy3016), Xiao Luo(xl1716)

11th February, 2018

Contents

1 Implementation Details **1**

2 Diagrams For Decision Trees **2**
2.1 Anger For Noisy and Clean 2
2.2 Disgust 3
2.3 Fear 4
2.4 Happiness 5
2.5 Sadness 6
2.6 Surprise 7
2.7 Data Preprocessing 7

3 Evaluation **8**
3.1 Clean Dataset Testing Result 8
3.2 Noisy Data Testing Result 11

4 Answer Questions **12**
4.1 Noisy-Clean Dataset Question 12
4.2 Ambiguity Question 13
4.3 Pruning Question 14

1 Implementation Details

In this coursework, we choose Python 3 as our implementation language and develop all of our own decision tree algorithms using Python with its basic libraries. Firstly, we load Matlab type data into Python by applying 'loadmat' function declared in scipy library. For training and evaluation, we split the data into two sets, the training data and testing data by 10 folds. For each set of data, we build six trees according to six emotions. Thus before building the decision trees, for each emotion, we use the function get_binary to transfer the multiple targets into binary targets. Finally, the decision tree is built.

Because of the recursive nature of the decision tree algorithm, a Tree class with three attributes and methods is applied to save all the information that is required to represent a single node. In non-leaf node class, 'op' is used to denote current attribute (AU 1-45) and kids are its kids nodes. For leaf node class, it will has one attribute which is prediction result. We use two different methods for choosing the best attribute for each node, which are ID3, and C4.5. The detailed analysis will be shown later. After finding the best attribute, the total sample will be divided into one set that has positive attribute and another with negative attribute. Afterwards, we can extent the decision tree by setting the best attribute into each of the branches until there is either no attribute left or the sample is empty or all the samples have the same binary target.

What is worth mentioning is that We do some data preprocessing to discard some useless data samples. This refer to those data with different attributes but same prediction results. We have proved this method could prevent our tree from extremely deep in some branches, and this will be shown in the visualization part.

To combine the six different trees into one tree, we tried three different methods. If there is just one tree predicting the positive result, nothing else needs to be done. If two or more trees produce positive values, our first strategy is randomly picking one as the final result. We also think it's a good idea to choose the tree that has the minimum depth, which can reduce the risk of over-fitting. Finally, the last strategy is to give preference to the balanced trees, which handle all possible values of the attributes equally.

After that, we use the predicted data we have generated to draw the confusion matrix and compute all the model estimation parameters. Moreover, we tried other algorithms like random forest to improve the performance of the decision tree

2 Diagrams For Decision Trees

2.1 Anger For Noisy and Clean

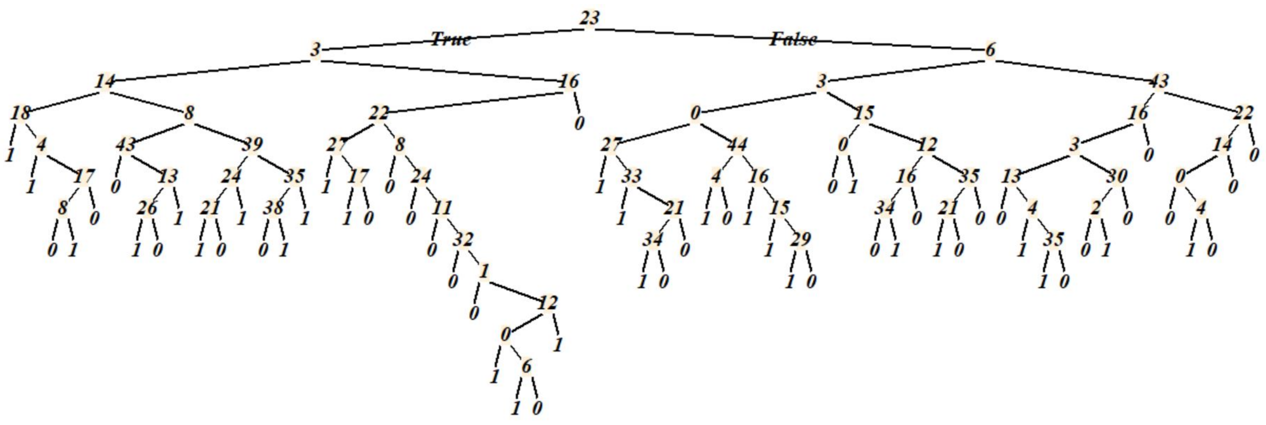


Figure 1: decision tree for anger emotion from clean data

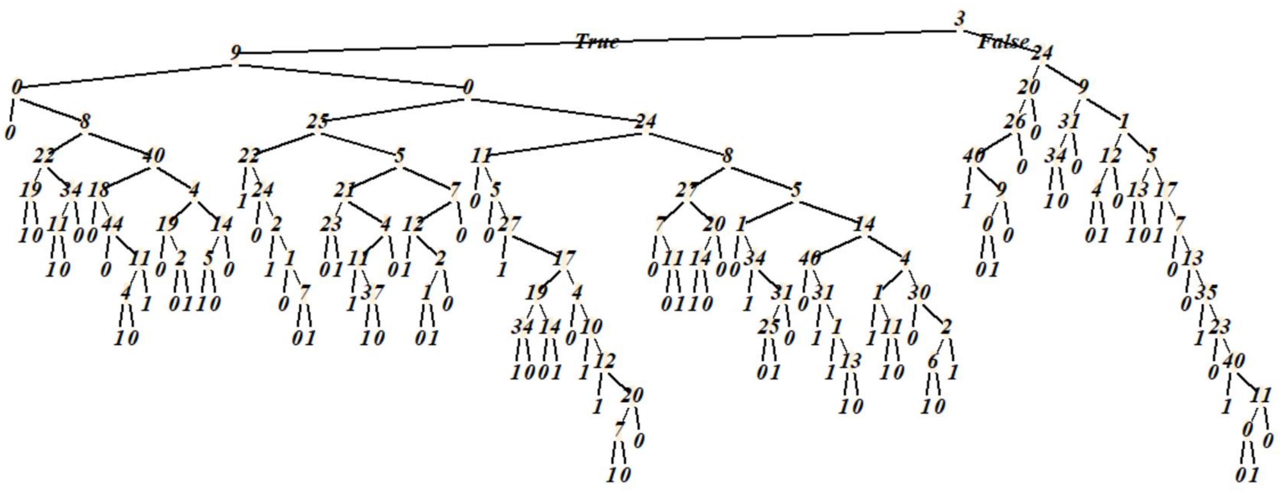


Figure 2: decision tree for anger emotion from noisy data

2.2 Disgust

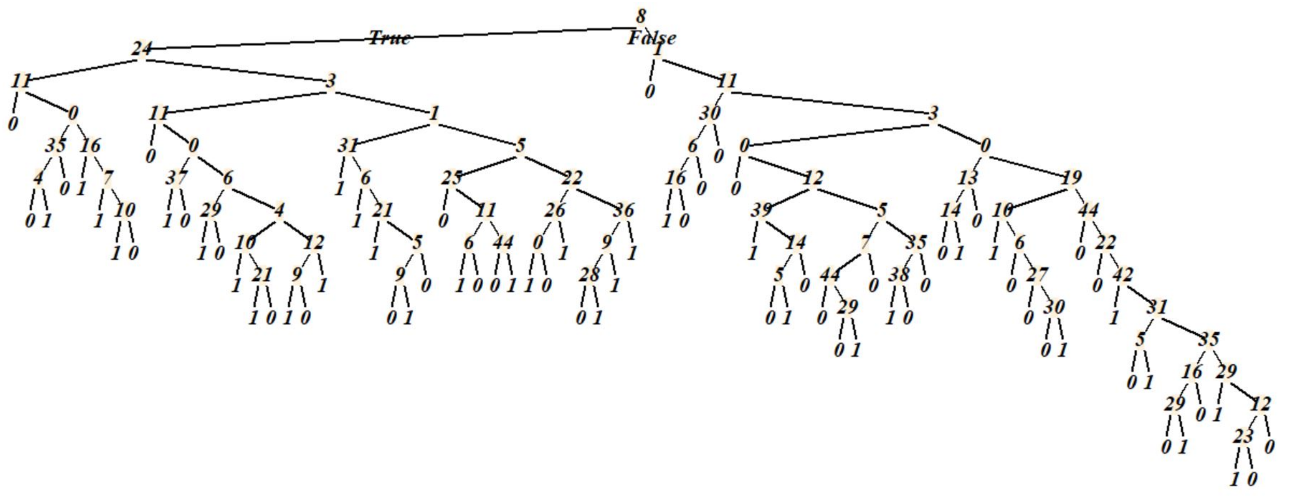


Figure 3: decision tree for disgust emotion from clean data

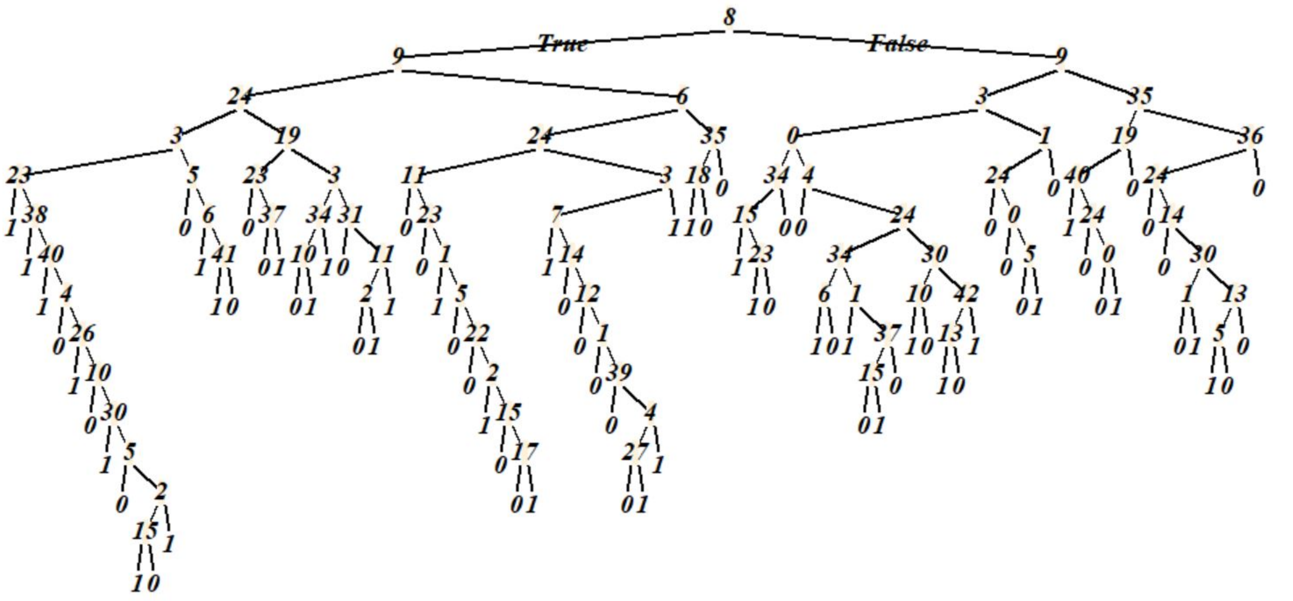


Figure 4: decision tree for disgust emotion from noisy data

2.3 Fear

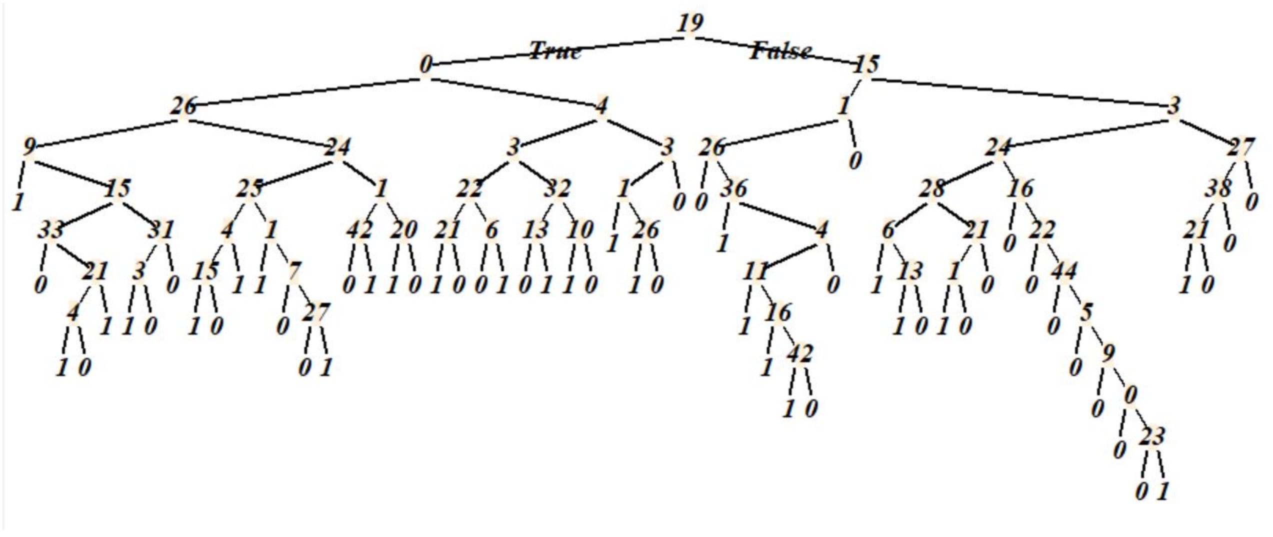


Figure 5: decision tree for fear emotion from clean data

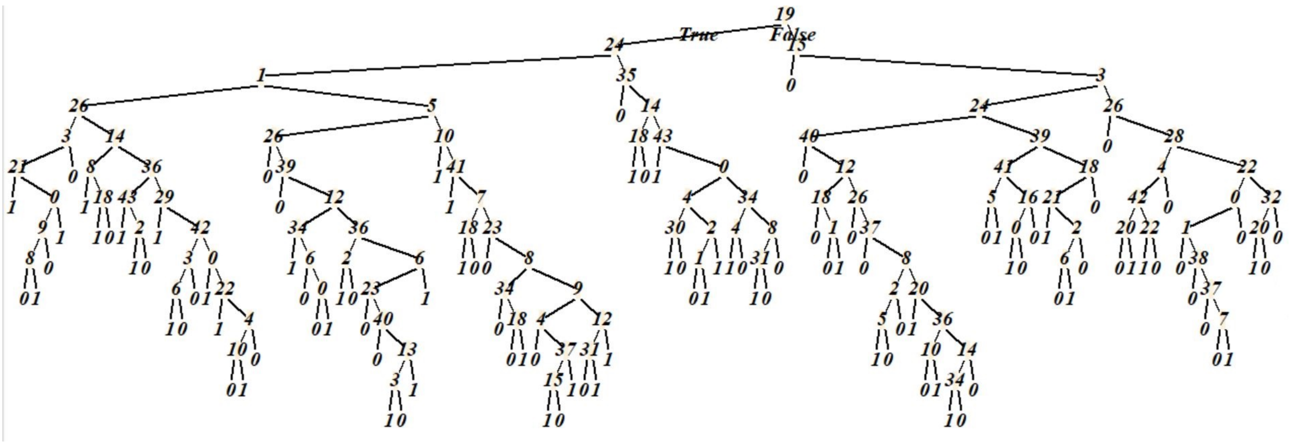


Figure 6: decision tree for fear emotion from noisy data

2.4 Happiness

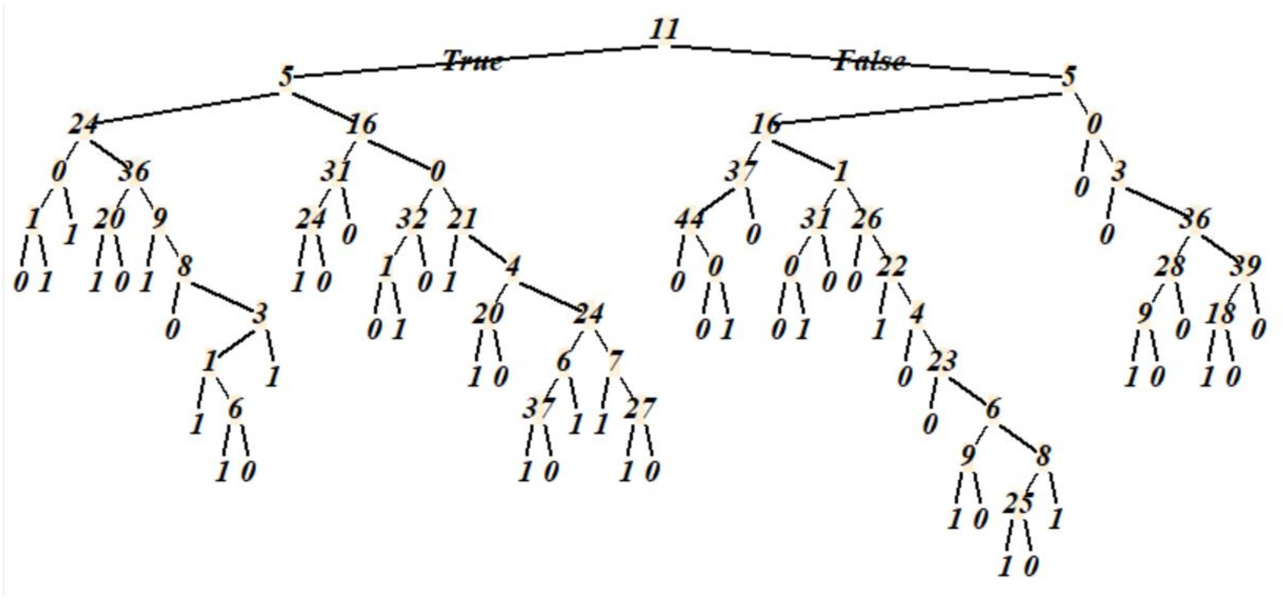


Figure 7: decision tree for happiness emotion from clean data

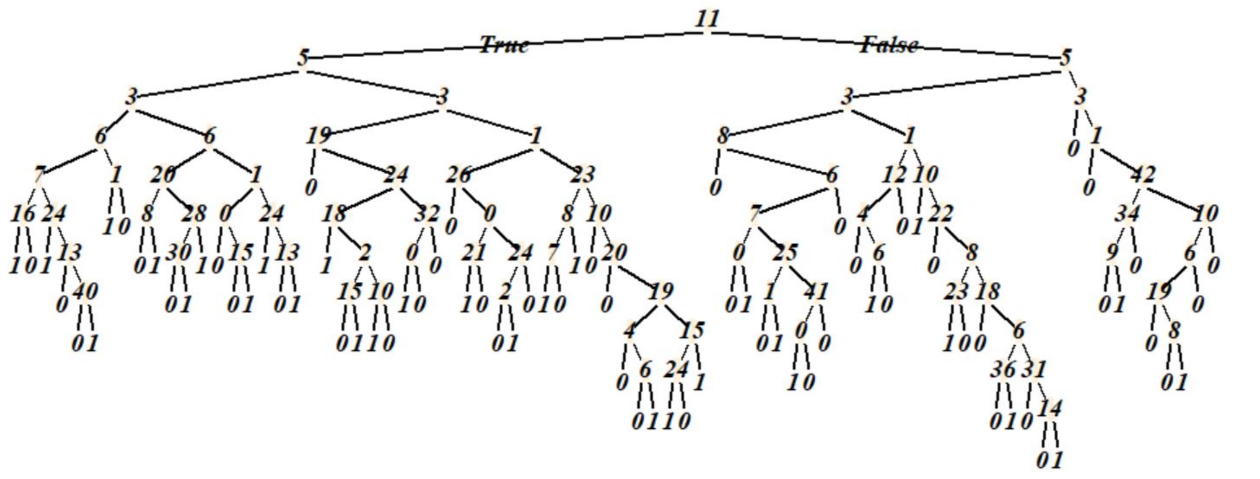


Figure 8: decision tree for happiness emotion from noisy data

2.5 Sadness

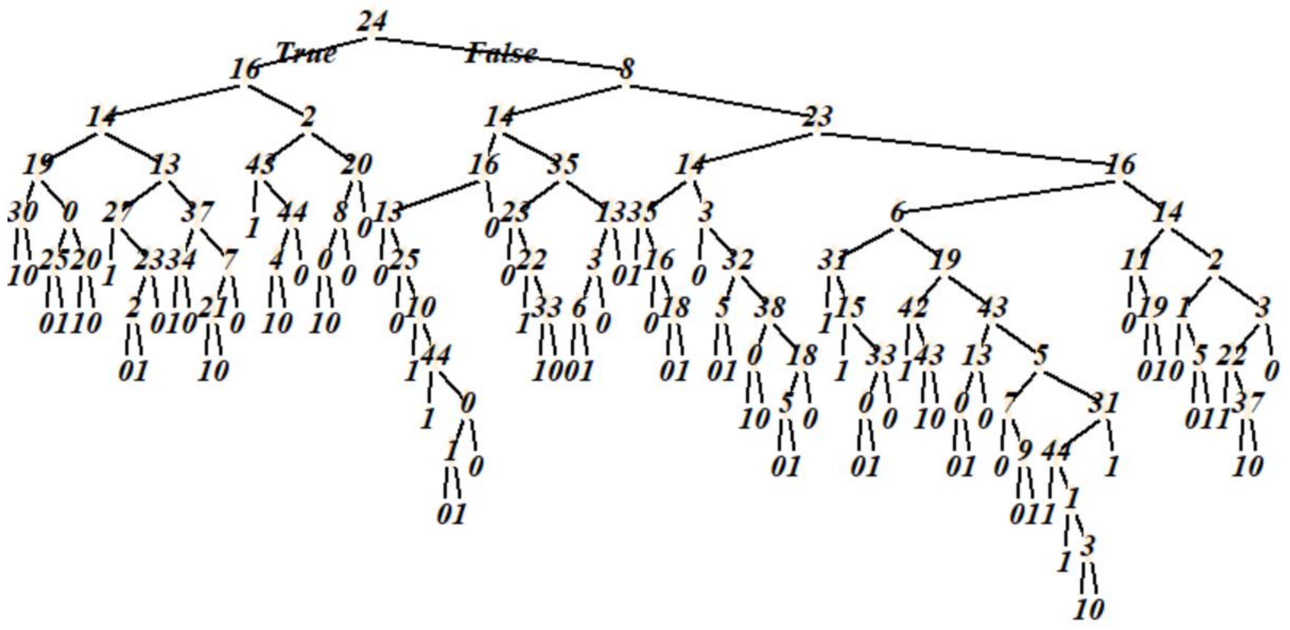


Figure 9: decision tree for sadness emotion from clean data

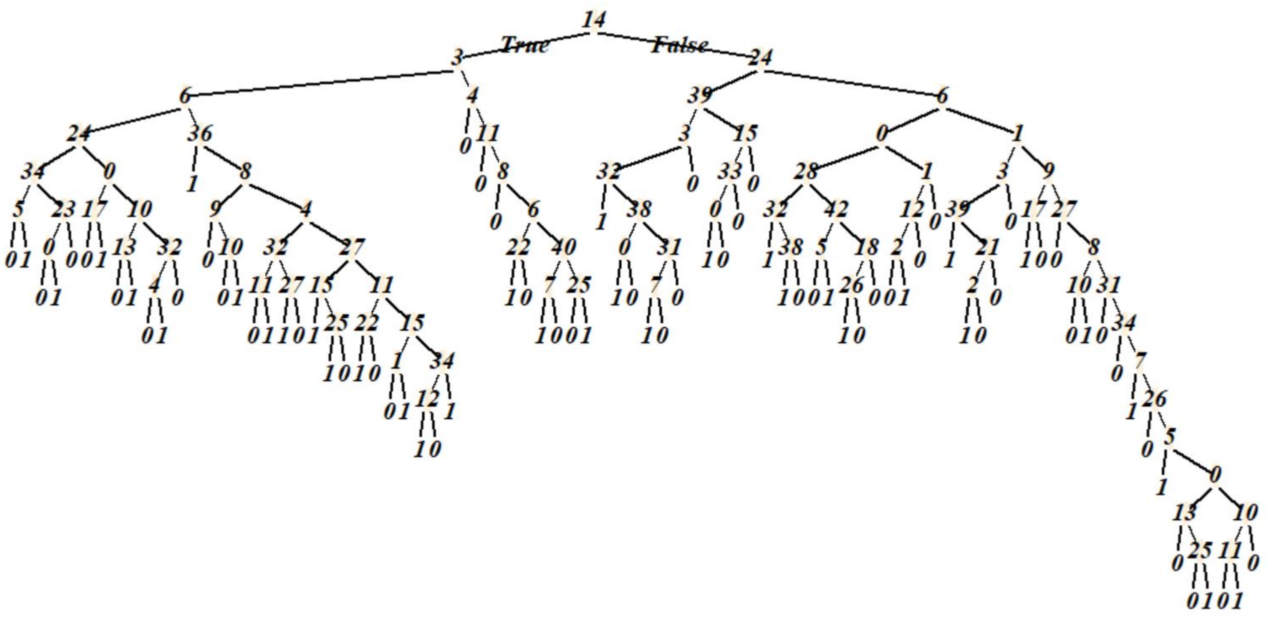


Figure 10: decision tree for sadness emotion from noisy data

2.6 Surprise

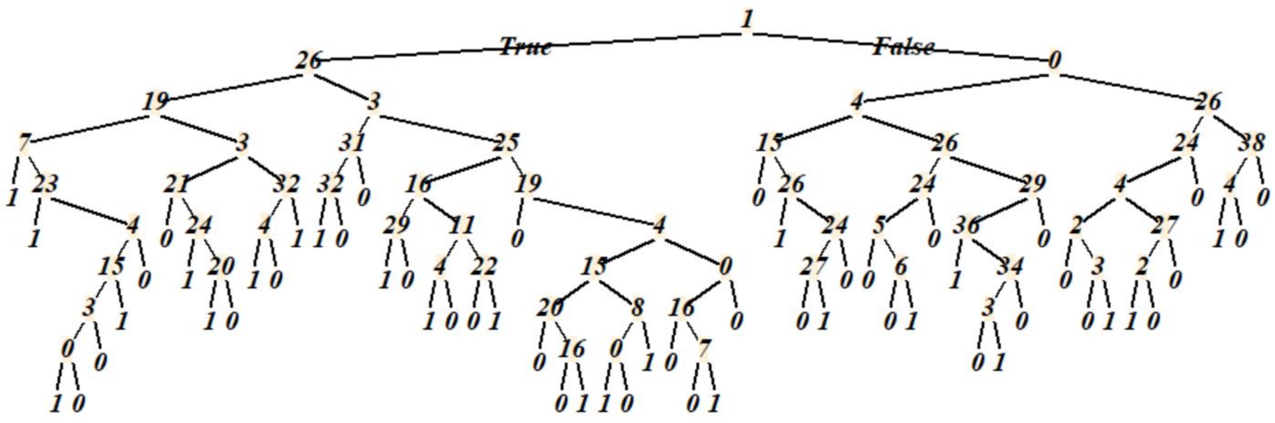


Figure 11: decision tree for surprise emotion from clean data

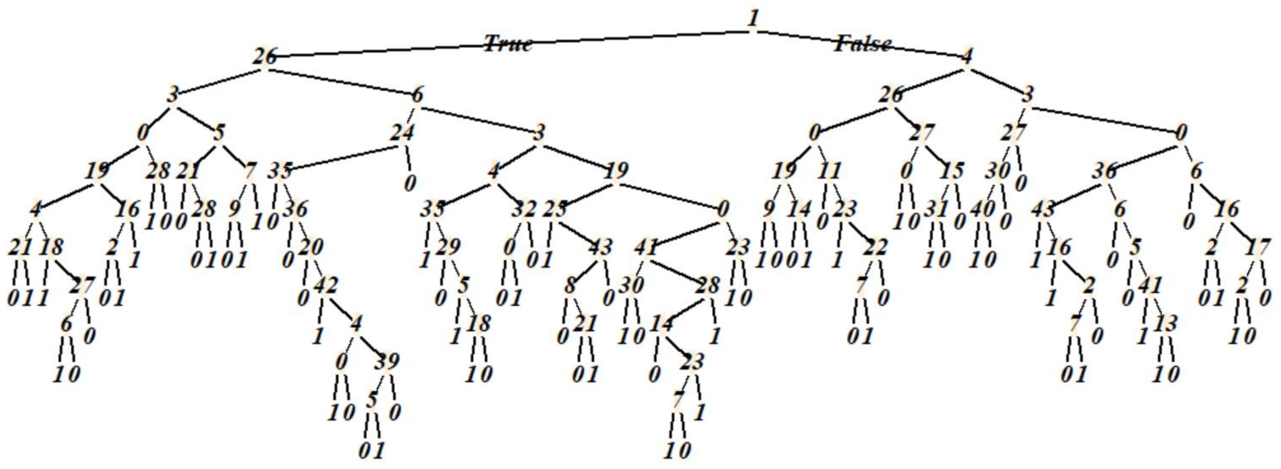


Figure 12: decision tree for surprise emotion from noisy data

2.7 Data Preprocessing

The basic idea of our data preprocessing is that we will check our data examples and their corresponding targets ahead of training the tree. If we find that some data have exactly same attributes but different targets (results), these samples will be regarded as not reasonable. If these samples mostly produce Result r_1 but only a few produce Result r_n, we will just force all these samples to produce the same results r_1. If the results just distribute randomly, we will discard all of them. The reason for this data preprocessing is to guarantee the quality of our data, including its accuracy, consistency and integrity. For example, it's possible that some errors of data collection have been made due to man-made manipulation or instruments' malfunctions. It can also happens when there exist some bias when obtaining the data. If we deal with these problems in advance, we can avoid some problems, such as making our tree too deep or not balanced. The comparison is illustrated in Figure 13. We can see that the bottom one is the training result without data preprocessing. The branch is the left is too deep and actually most of them are useless for our prediction and can even lead to some confusion and bias.

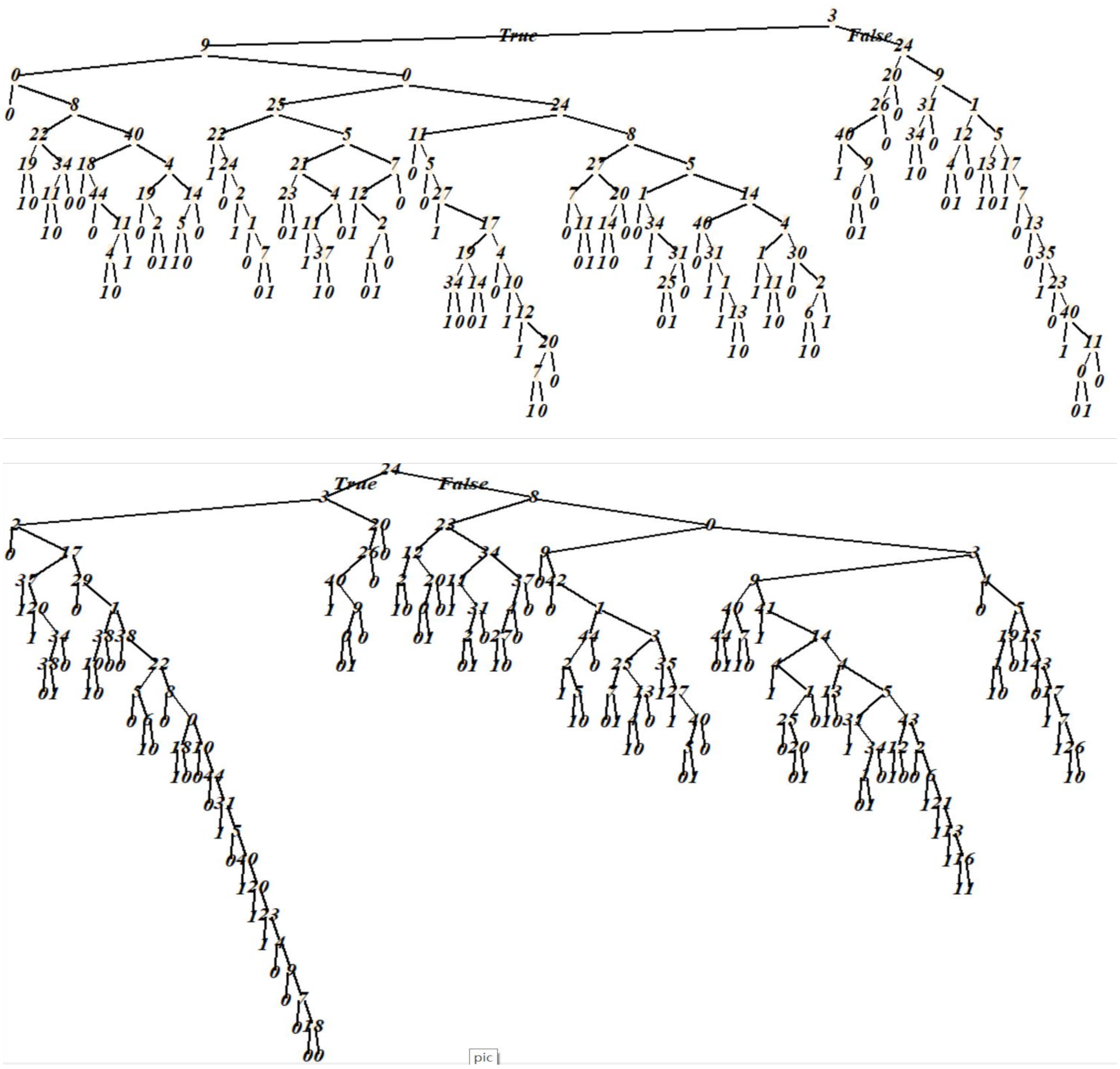


Figure 13: Comparison between anger emotion tree derived from noisy data with and without data preprocessing

3 Evaluation

The 10-fold validation was performed both on the noisy and clean dataset and the results are presented as below. Such results were obtained by adopting the random selection method to ensure that only one emotion was assigned to a certain example. For each fold validation, there are 100 testing examples and the remaining (around 900) are training dataset. The average confusion matrix is obtained by summing up the confusion matrix of each fold and then divided by 10. The average error is calculated in the same way, by averaging errors of each fold. Then the classification rate is obtained by subtracting the average error from 1.

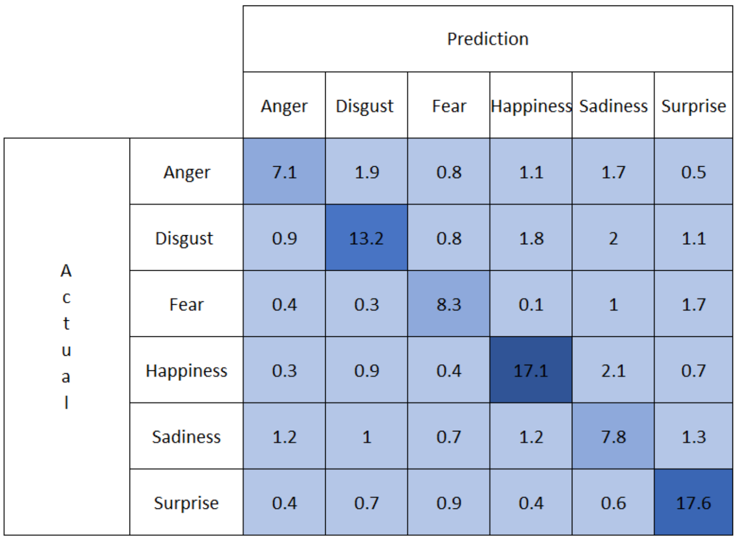
3.1 Clean Dataset Testing Result

Average Classification Rate:

$$Classification\ Rate = \frac{720}{1004} = 72\% \tag{1}$$

The average classification rate gives a general idea about how well the built decision trees can classify the testing data. The 72% classification rate suggests that out of 100 testing examples, there would be 72 correctly classified. To know the detailed classification performance of certain emotion, a confusion matrix is needed.

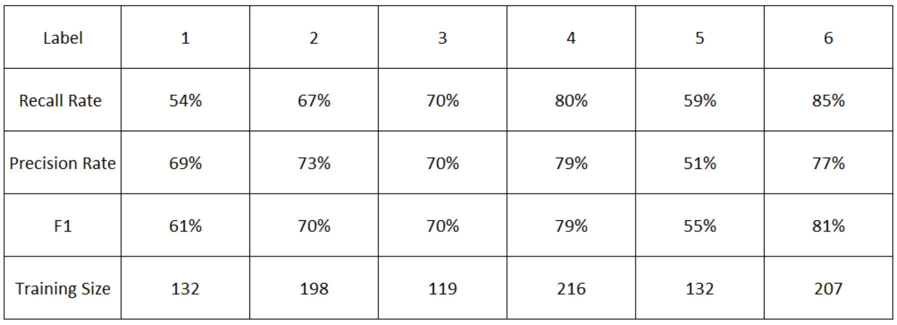
Confusion Matrix:
Figure 14 shows the average confusion matrix for the clean dataset. Cells with darker colour have a larger value within. This matrix reveals that emotions including disgust, happiness and surprise (label 2, 4, 6) have much more correctly classified instances than others. However, it does not imply that these three emotions are better classified than others because those emotions with more testing instances are more likely to have more correctly classified examples. As shown in Figure 15, the testing and training dataset size for these 3 labels are larger than others. To further investigate the testing result, the recall rate, precision rate and F1 are needed.



		Prediction					
		Anger	Disgust	Fear	Happiness	Sadiness	Surprise
Actual	Anger	7.1	1.9	0.8	1.1	1.7	0.5
	Disgust	0.9	13.2	0.8	1.8	2	1.1
	Fear	0.4	0.3	8.3	0.1	1	1.7
	Happiness	0.3	0.9	0.4	17.1	2.1	0.7
	Sadiness	1.2	1	0.7	1.2	7.8	1.3
	Surprise	0.4	0.7	0.9	0.4	0.6	17.6

Figure 14: Confusion Matrix for the clean dataset

Recall Rate, Precision Rate and F1:



Label	1	2	3	4	5	6
Recall Rate	54%	67%	70%	80%	59%	85%
Precision Rate	69%	73%	70%	79%	51%	77%
F1	61%	70%	70%	79%	55%	81%
Training Size	132	198	119	216	132	207

Figure 15: Recall Rate, Precision Rate and F1 for the clean dataset

In Figure 15, the training size of certain label is the total number of examples in the training dataset classified as this label. Precision rate is a measurement of how well a certain tree can distinguish the emotion it tests with other emotions. Recall rate is a measurement of how well the group of trees recognize a certain emotion. If certain emotion has larger testing data size, it tends to have a larger precision rate because even if the tree of this emotion has lower accuracy, the number of correctly classified instances may still dominant the whole column in the confusion matrix. If certain emotion has larger training data size, it tends to have larger recall rate because all the trees will have more training data to classify this emotion and thus the model built will fit this emotion better. However, that is not always the case because too large training size sometime will cause overfitting.

Because the training data and testing data are drawn randomly from the same sample space, the portions of these two types of data for certain emotion are both proportional to the its occurrence in the who dataset. From Figure 15 we can find the emotion 4 and 6 are much better classified than other emotions and apparently these two emotions have the largest dataset size. One exception is the label 2, even if it has a quite large training data size, the classification accuracy is relative low. This could result from overfitting. Figure 16 and 17 are the plots for all the testing results for 10-fold validation. Each point represents the F1 value for certain label in certain fold and there are 60 points in total. The x-axis in Figure 16 represents the training data size for the corresponding emotion of F1 and that of figure 17 represents the testing data size. Although not quite apparently for Figure 16 (may due to overfitting), we can still see that a large training dataset size can lead to a larger F1. The relation between testing dataset size and F1 is more apparently indicated in Figure 17.

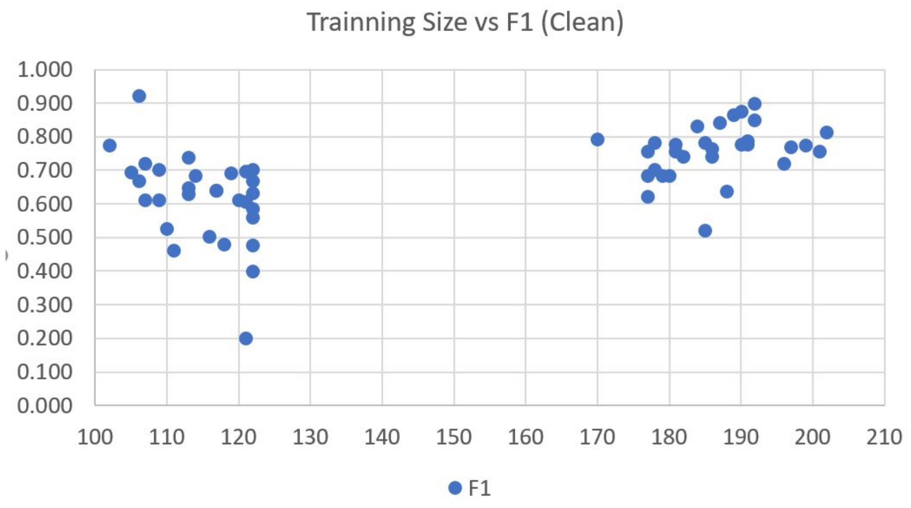


Figure 16: Training Data Size vs F1 for all 10-Fold Test Example

Recall Rate, Precision Rate and F1:

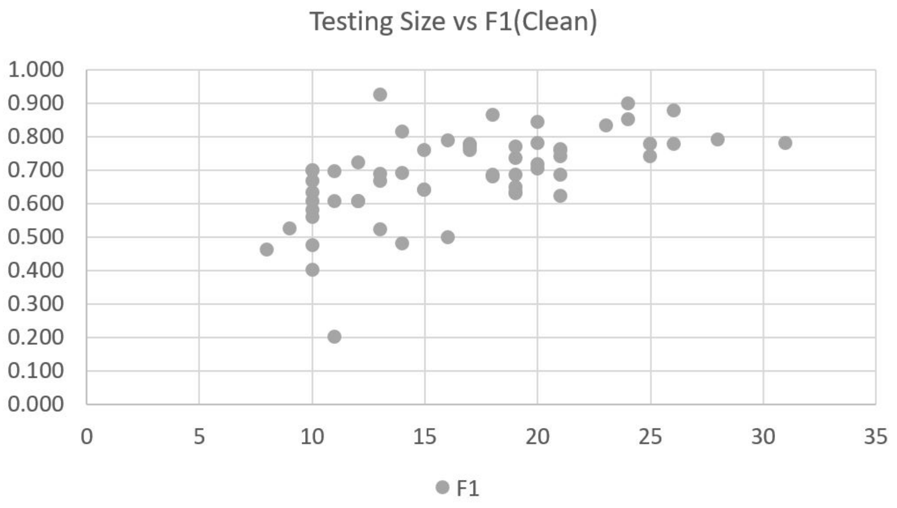


Figure 17: Testing Data Size vs F1 for all 10-Fold Test Example

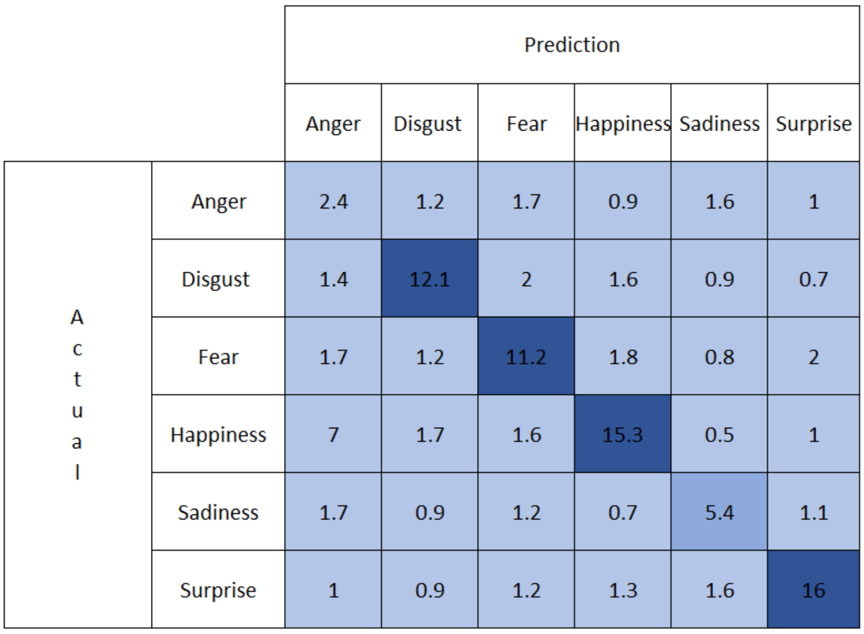
3.2 Noisy Data Testing Result

Average Classification Rate:

$$Classification\ Rate = \frac{620}{1001} = 62\% \tag{2}$$

Compared with 72% classification rate of performance on clean dataset, the 62% classification of noisy dataset indicates the decision tree algorithm designed has better performance on clean data.
Confusion Matrix:

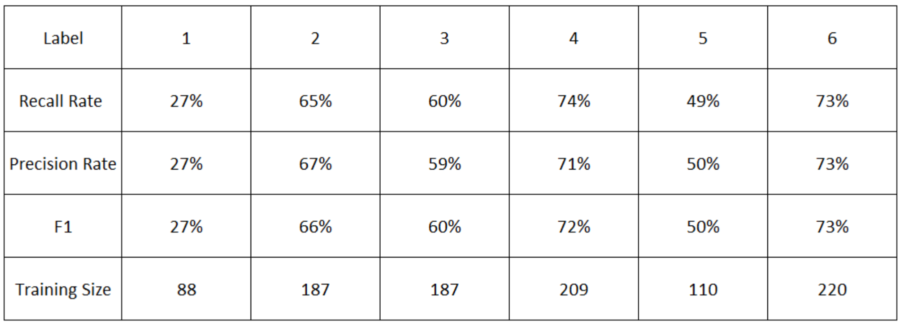
Figure 18 shows the average confusion matrix for the noisy dataset. This matrix reveals that emotions including disgust, happiness and surprise (label 2, 3, 4, 6) have much more correctly classified instances than others. That is because the testing and training dataset size for these four labels are larger than others as will be shown later in Figure 19.



		Prediction					
		Anger	Disgust	Fear	Happiness	Sadiness	Surprise
Actual	Anger	2.4	1.2	1.7	0.9	1.6	1
	Disgust	1.4	12.1	2	1.6	0.9	0.7
	Fear	1.7	1.2	11.2	1.8	0.8	2
	Happiness	7	1.7	1.6	15.3	0.5	1
	Sadiness	1.7	0.9	1.2	0.7	5.4	1.1
	Surprise	1	0.9	1.2	1.3	1.6	16

Figure 18: Confusion Matrix for the Noisy Dataset

Recall Rate, Precision Rate and F1:



Label	1	2	3	4	5	6
Recall Rate	27%	65%	60%	74%	49%	73%
Precision Rate	27%	67%	59%	71%	50%	73%
F1	27%	66%	60%	72%	50%	73%
Training Size	88	187	187	209	110	220

Figure 19: Recall Rate, Precision Rate and F1 for the Noisy Dataset

From Figure 19 we can find the emotion 4 and 6 are better classified than other emotions and apparently these two emotions have the larger dataset size than others. The classification performance for the label 2 and 3 is not as good as the label 4 and 6 despite of generally the same dataset size. The classification performance for the label 1and 5 are particularly bad which is very likely due to the small training size which leads to insufficiently training of these labels. For the noisy dataset, the relations between F1 and Training Size as well between F1 and Testing size is much more apparent as shown in Figure 20 and 21.

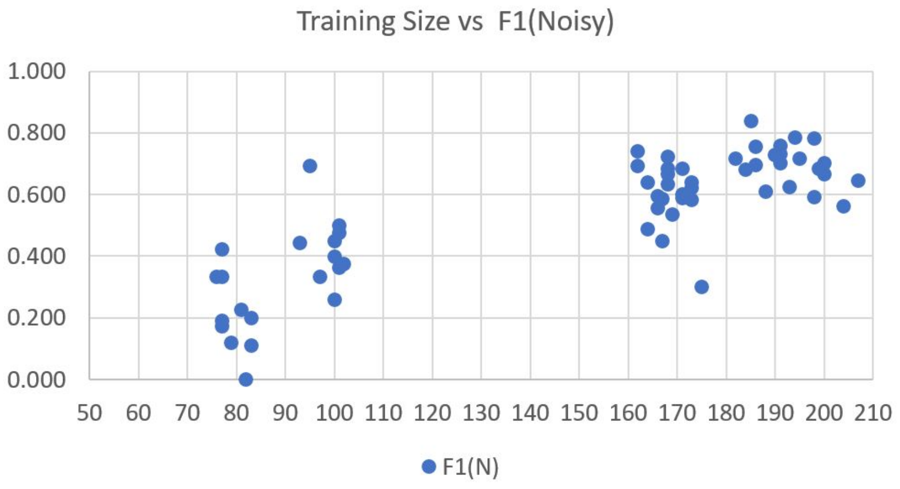


Figure 20: Training Data Size vs F1 for all 10-Fold Test Example (Noisy Data)

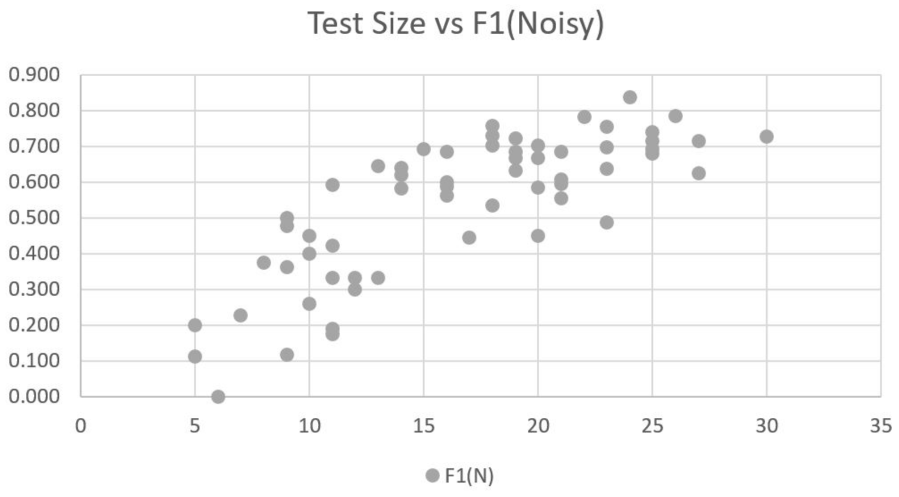


Figure 21: Testing Data Size vs F1 for all 10-Fold Test Example (Noisy Data)

4 Answer Questions

4.1 Noisy-Clean Dataset Question

$$Noisy\ Classification\ Rate = \frac{620}{1001} = 62\% \tag{3}$$

$$Clean\ Classification\ Rate = \frac{720}{1004} = 72\% \tag{4}$$

Trees trained with the noisy dataset has a worse classification performance as indicated by the classification rate. That is because there are incorrectly labelled data in the noisy dataset and the training algorithm would try to fit these data which will cause the built trees to less fit the correctly labelled data. In figure 22, each point represents the F1 value for certain label in certain fold of 10-fold validation and 60 points in total. It shows that trees perform between on clean dataset for all the labels. It implies that the trees have better performance on clean dataset for all the labels than on noisy dataset.

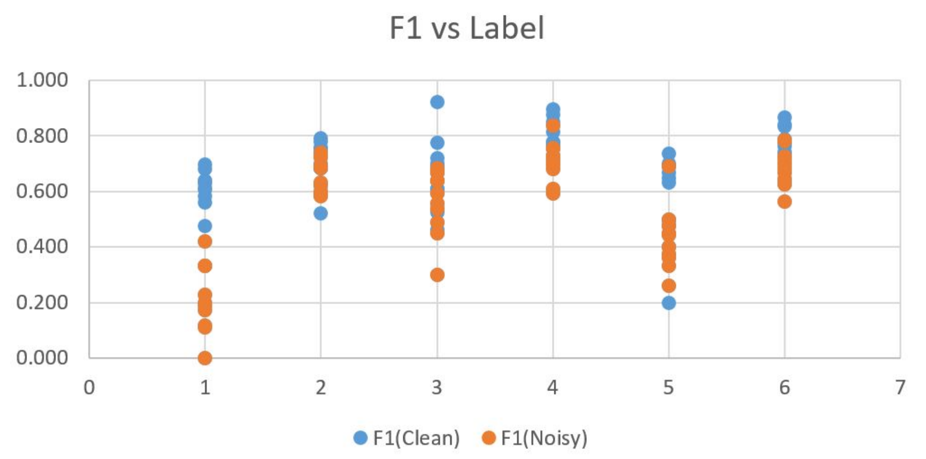
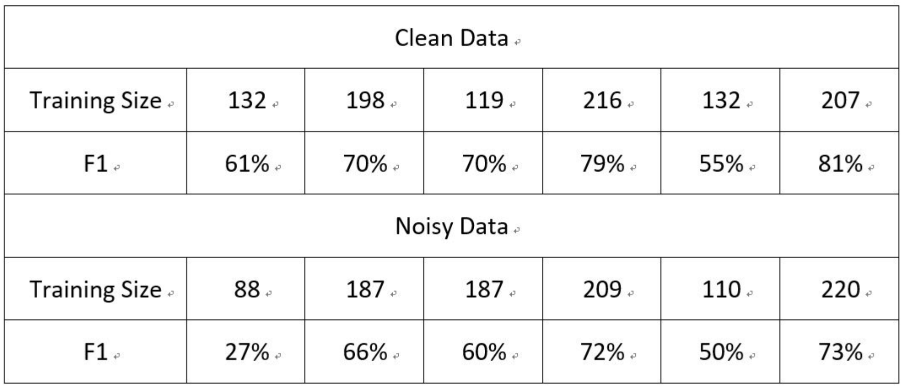


Figure 22: F1 for Clean and Noisy Dataset for Certain Label of All 10-fold Test



Clean Data						
Training Size	132	198	119	216	132	207
F1	61%	70%	70%	79%	55%	81%
Noisy Data						
Training Size	88	187	187	209	110	220
F1	27%	66%	60%	72%	50%	73%

Figure 23: The Clean dataset and Noisy dataset result comparison

Another observation is trees trained with noisy and clean dataset both have best performance on label 4 and 6 as indicated by Figure 23. One reason for this might be the label 4 and 6 have the largest training and testing data size in both datasets. Another reason can be these two emotions have distinct features which makes them easier to classified than others. Label 1 and Both in noisy and clean dataset, label 1 and 5 have the worse performance which may due to their smaller dataset or difficulty in extracting its feature pattern. Another reason could be there are more noisy data for these two labels. Label 2 and 3 have medium performance in both two datasets. In the clean data, even label 3 has the smallest training dataset size, it performs much better than label 1 and 5 and the performance of label 3 is the same as label 2 even the latter one has much larger dataset. In the noisy dataset, despite the large dataset of label 2 and 3, they do not have the same performance as label 4 and 6. The reason for this result might because these two labels are more likely to overfit the training model.

4.2 Ambiguity Question

Each example needs to get only a single emotion assigned to it, between 1 and 6. To achieve that when multiple labels are assigned to certain example, we use four strategies:

1. Random Selection: randomly select one label from assigned one as the prediction.
2. Max Tree Depth: choose the label whose tree has maximum depth
3. Min Tree Depth: choose the label whose tree has minimum depth
4. Max Training Size: choose the label whose training data size is the largest

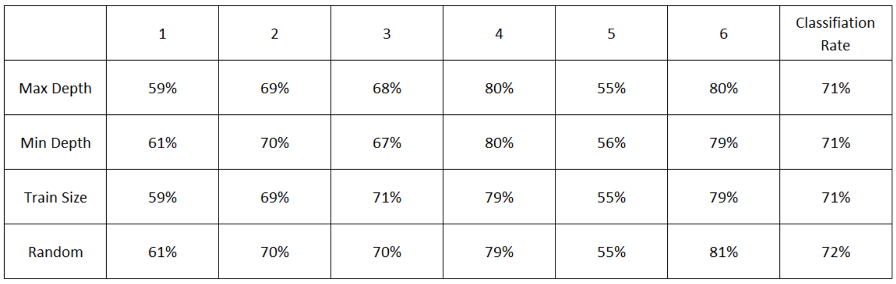
The advantages and disadvantages of the four strategies are as follows: For strategy 1, random selection is probably the most intuitive way of dealing with multiple choice question, and the easiest way regarding implementation. The strategy makes sure that given multiple emotions assigned to an example, each emotion has equal possibility to be the final result. However, if confidence value of emotions assigned to the same result different from each other, Random Selection may result in a biased estimation.

For strategy 2, the label with maximum tree depth may provide us with more specific information, and might sometimes generate more accurate prediction. Nevertheless, the presence of noisy data may jeopardize the accuracy. For strategy 3, it comes from the dictum Greatness in Simplicity. If a result can be categorized only after a few tree nodes, it may probably be a typical example. Given that our facial emotions are controlled by the specific muscular movement which is limited in number, Min Tree Depth may be believed to perform well.

Another disadvantage of strategy 2 and 3 is that they have to do random selection if the trees has equal depth, but the possibility of the situation is relative low.

For strategy 4, as discussed before, the label with large training size are more likely to have a better classification performance. Therefore it is more believable that the label with larger training size tends to be more correct when giving a positive result.

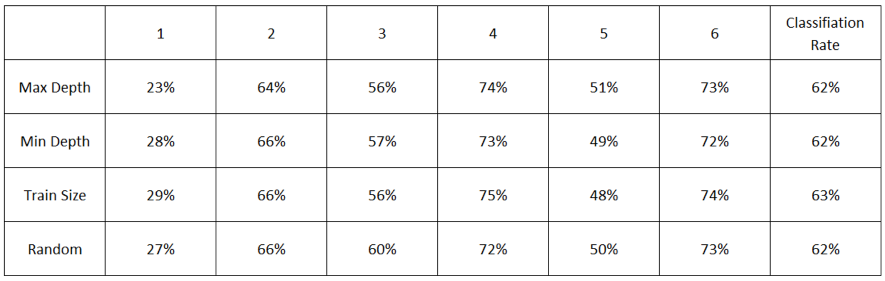
Result:
Clean:



	1	2	3	4	5	6	Classifiation Rate
Max Depth	59%	69%	68%	80%	55%	80%	71%
Min Depth	61%	70%	67%	80%	56%	79%	71%
Train Size	59%	69%	71%	79%	55%	79%	71%
Random	61%	70%	70%	79%	55%	81%	72%

Figure 24: The Clean dataset and Noisy dataset result comparison

Noisy:



	1	2	3	4	5	6	Classifiation Rate
Max Depth	23%	64%	56%	74%	51%	73%	62%
Min Depth	28%	66%	57%	73%	49%	72%	62%
Train Size	29%	66%	56%	75%	48%	74%	63%
Random	27%	66%	60%	72%	50%	73%	62%

Figure 25: The Clean dataset and Noisy dataset result comparison

The classification rates of both clean and noisy data show no significant difference: out of 1000 testing data (100 for each fold), there are 179 examples in noisy dataset and 148 examples in clean dataset assigned with multiple label during testing. Since the ratio of multiple assigning is relatively low, the difference between the four methods seems less obvious. Therefore, we cannot conclude whether our findings are consistent with what we described above, and because each strategy cannot be theoretically proved have dominant performance over one another, and the cost of the strategies is similar, choosing either of them shall be fine in this case.

4.3 Pruning Question

Pruning_example function mainly plots the relationship between pruning cost and decision tree size. The red line and blue line in the graph, represents training error and validation error, respectively. The function generates a decision tree in the first line, followed by ten-fold cross validation and resubstitution test. Then it calculates the minimum-cost tree separately according to estimated best level of pruning generated in the two testing. Finally, it plots the estimated cost of each tree size and marks the optimal tree size with square.

The red line, indicating the resubstitution method, shows that the cost decreases sharply at first and slowly to zero afterwards when tree nodes increases. That is because the cost of decision tree represents the sum of misclassification in the test, while resubstitution method testing is actually based on the same training data. Therefore, given that minparent equals 1, the training error decreases steadily to zero.

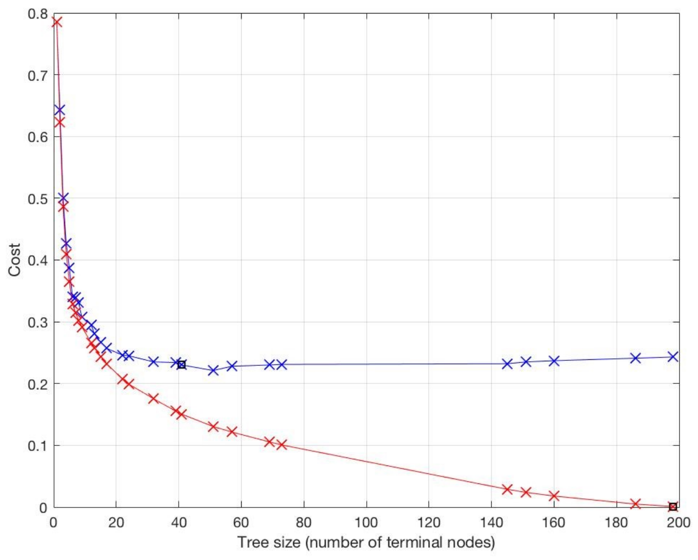


Figure 26:

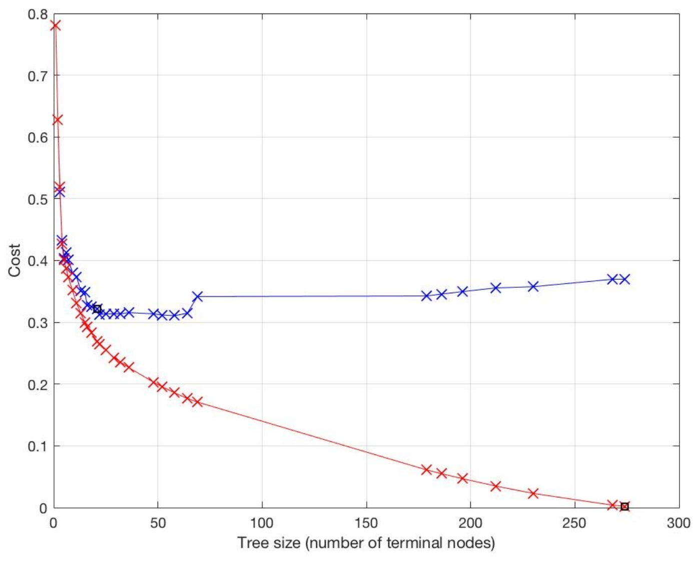


Figure 27:

However, the blue line sees that the cost decreases at first, and increases as the tree grows deeper, because the 10-fold cross validation uses different testing data, and overfitting occurs when terminal nodes increase.

Obviously, the validation error is greater than the training error, so the red line should locate below the blue line. Since noisy data incur some abnormal value, it is easy to generate more branches than clean data, and the cost of decision tree should be higher than that of clean data.

Based on the graph, the optimal tree size is 41 for clean data, and 21 for noisy data in 10-fold cross-validation, respectively.