Imperial College London

395 Machine Learning
Assessed Coursework

Decision Trees

Member:
Jiaxin Chen (CID 01367151)
Jiayue Shen (CID 01370058)
Yuyang Xu (CID 01395216)
Suampa Ketpreechasawat (CID 01401906)

February 13, 2018

Contents

List of Figures

List of Tables

Chapter 1

Implementation Details

1.1 Decision Trees Construction

Firstly, the 6 decision trees, each representing one specific emotion, are constructed with the underlying ID3 algorithm using entire clean data set. The selection of attribute for each node depends on the value of information gain, which could be defined as follows:

$$Gain(attribute) = I(p, n) - Remainder(attribute)$$

where:

$$I(p, n) = -\frac{p}{p+n} log_2(\frac{p}{p+n}) - \frac{n}{p+n} log_2(\frac{n}{p+n})$$

$$Remainder(attribute) = \frac{p_0+n_0}{p+n} I(p_0, n_0) + \frac{p_1+n_1}{p+n} I(p_1, n_1)$$

The attribute which yields the highest information gain, is selected for one particular node. In case that there is an existence of multiple attributes with highest gain, the node's attribute will be randomly chosen from one of those. For each branch, this process will continue unless it reaches the point where the calculated entropy equals to 0 or all attributes are taken into account. Moreover, we have also included threshold factor for our baseline decision trees in order to avoid too deep depth and overfitting. If the ratio between the positive examples and negative examples are too low or too high (which means we have too many negative outputs and only few positive outputs [2 positive outputs and 98 negative outputs] or too many positive outputs and few negative outputs [2 negative outputs and 98 positive outputs]), then we stop growing our tree and end a node with the majority class.

In order to improve the efficacy of baseline decision tree model, we have also applied the concept of "feature selection". The intuition behind this implementation is that feature selection reduces model's complexity and run-time consumed for further analysis, but retains the same or even better level of accuracy. Before constructing the decision tree, we will select the most meaningful attributes and include only those attributes for training in order to reduce the possibility of overfitting and improve prediction performance on unseen dataset. As previously mentioned, what the conventional decision tree learning algorithm does is that it trains decision trees to classify 6 emotions based on the information gain of every single attribute. However, 45 attributes are considered too many for classifying each individual emotion. For instance, when we try to distinguish Happiness emotion from others, it should simply require only 2 or 3 action units, in other words 2 or 3 main attributes, together with some auxiliary attributes, regarding to the fact that not all of 45 attributes are related for one specific emotion. Indeed, an attempt to utilize all 45 attributes to train every single tree eventually lead to unnecessarily increasing the depth of the tree, and subsequent overfitting problem. Therefore, our aim is to eliminate those attributes which are considered redundant. The way we can perform this feature selection is to use random forest. Random forest is the term referring to a number of small decision trees created by the subsets of attributes. While training those trees, we could then evaluate how each attribute reduce "impurity" of the trees. By averaging the results from every tree, the attributes could be ranked accordingly (for more details, please see [1] [2]). The 6 baseline decision trees and 6 decision trees with feature selection could be visualised as follows.

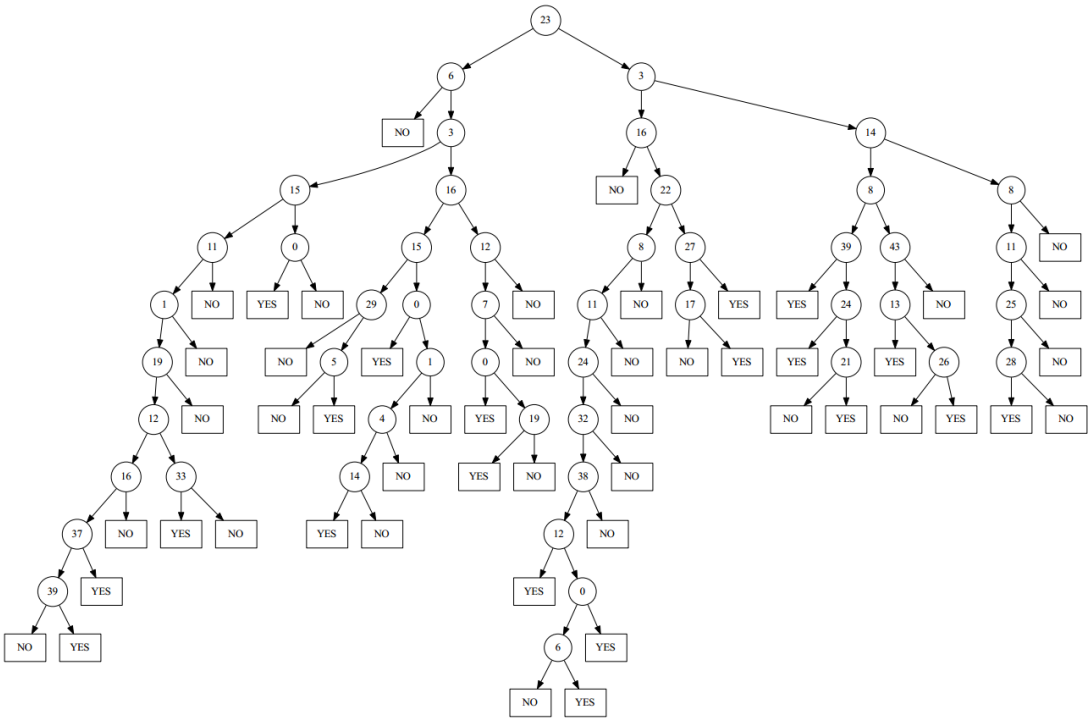

Figure 1.1: Baseline decision tree representing Anger

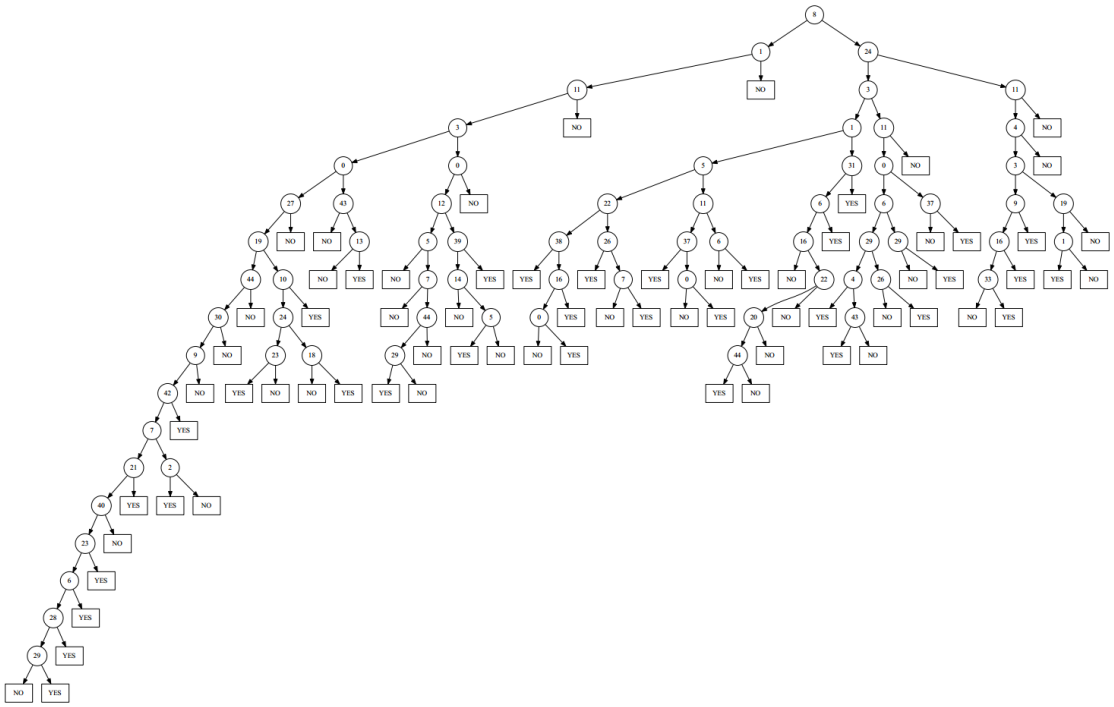

Figure 1.2: Baseline decision tree representing Disgust

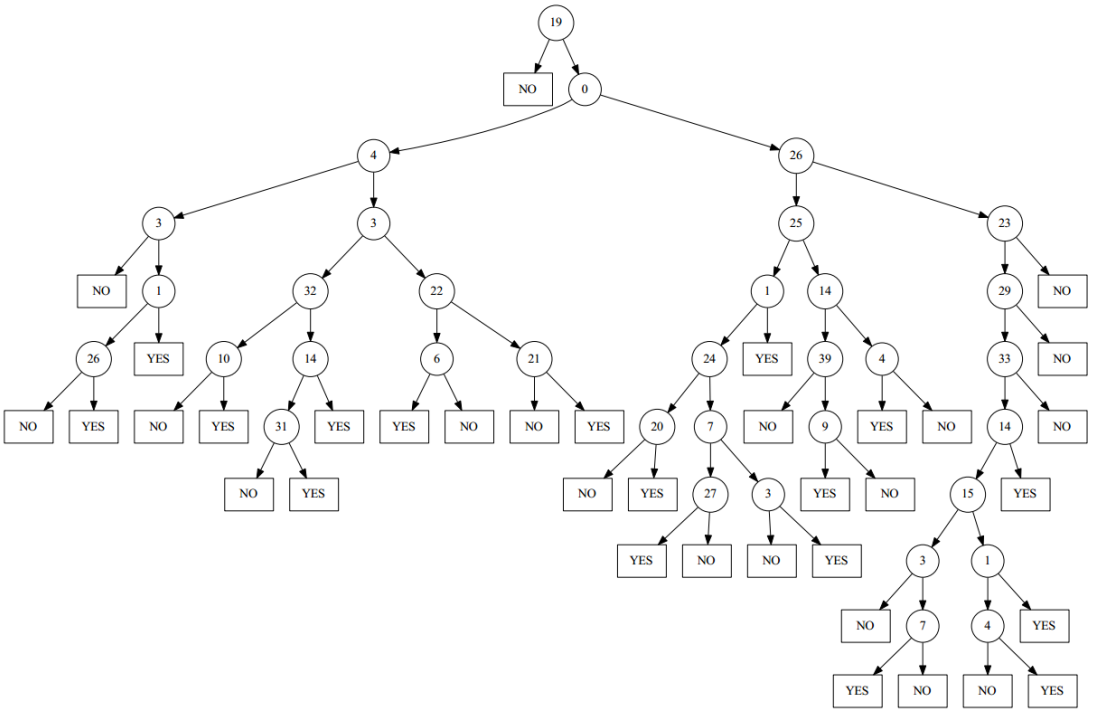

Figure 1.3: Baseline decision tree representing Fear

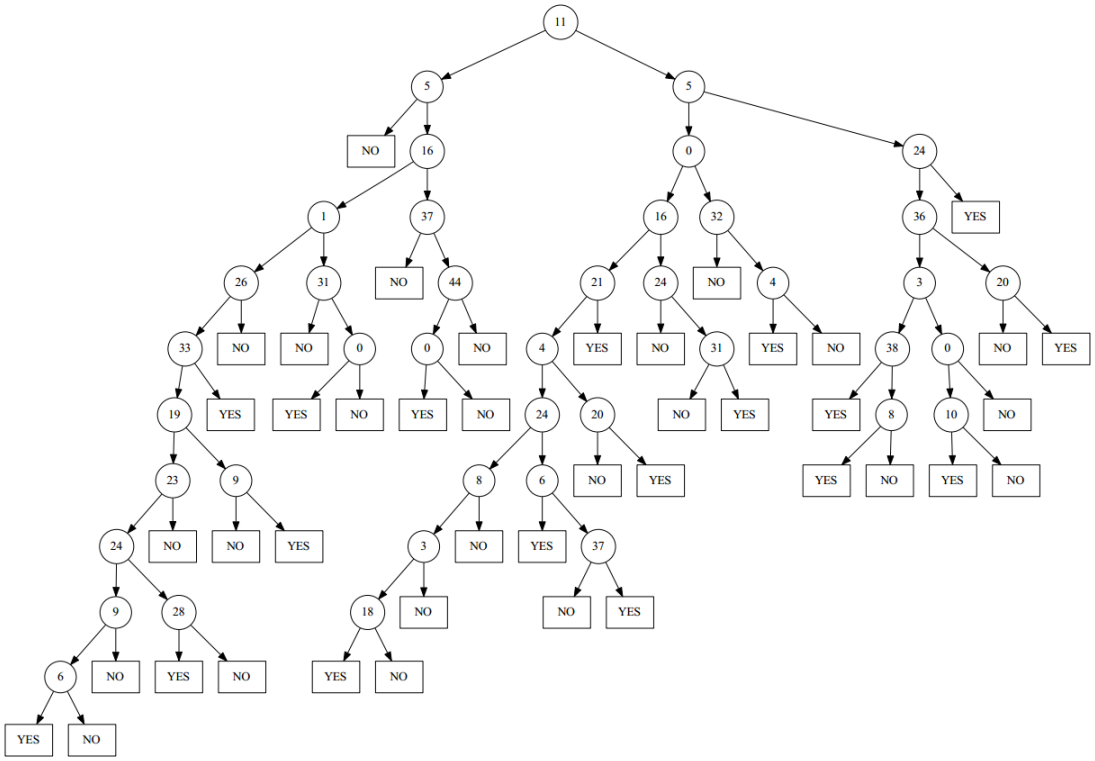

Figure 1.4: Baseline decision tree representing Happiness

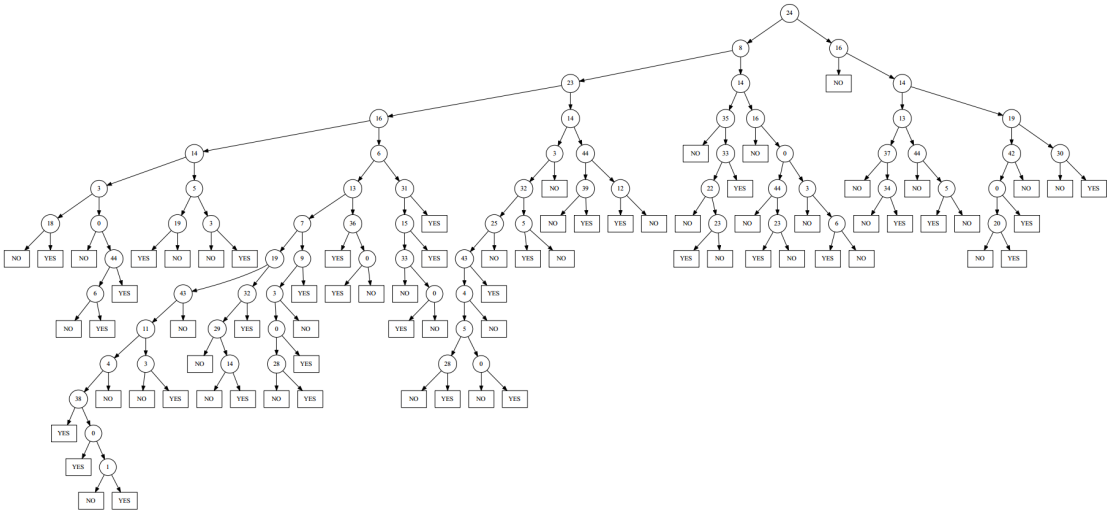

Figure 1.5: Baseline decision tree representing Sadness

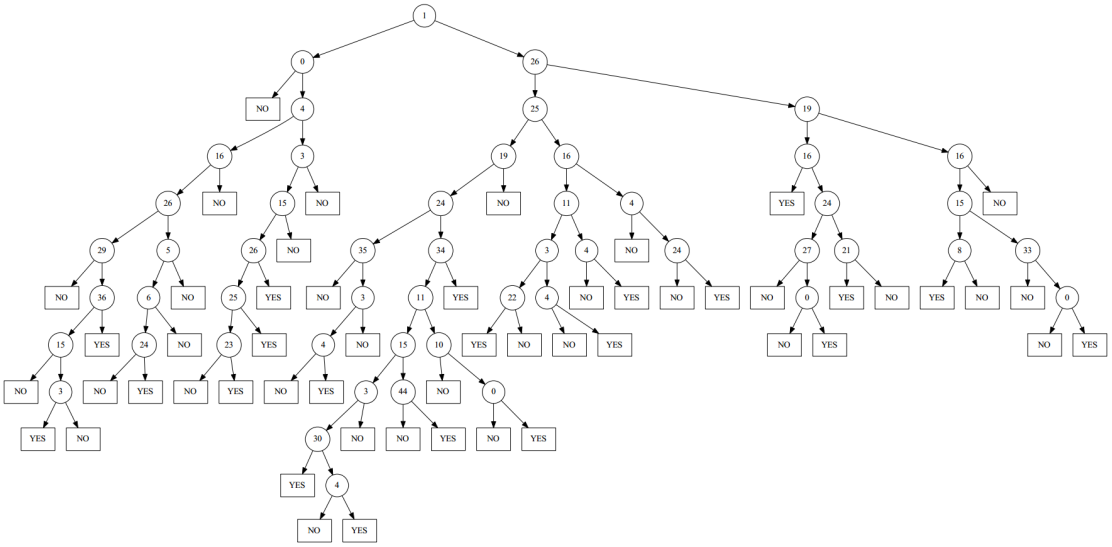

Figure 1.6: Baseline decision tree representing Surprise

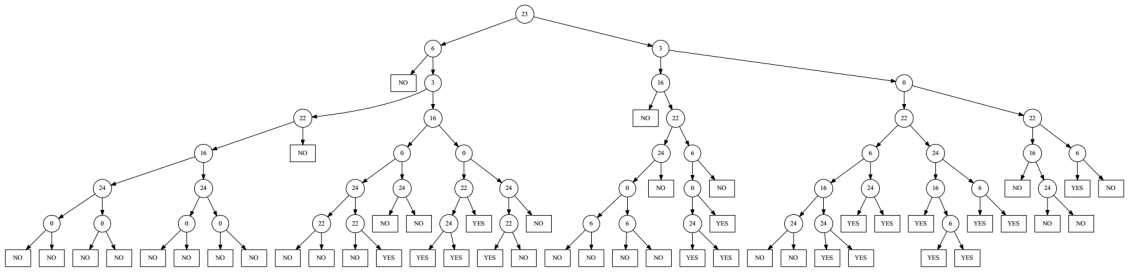

Figure 1.7: Decision tree with feature selection representing Anger

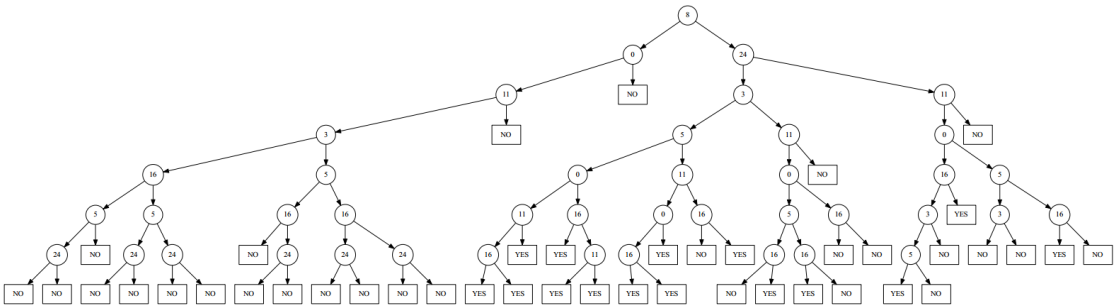

Figure 1.8: Decision tree with feature selection representing Disgust

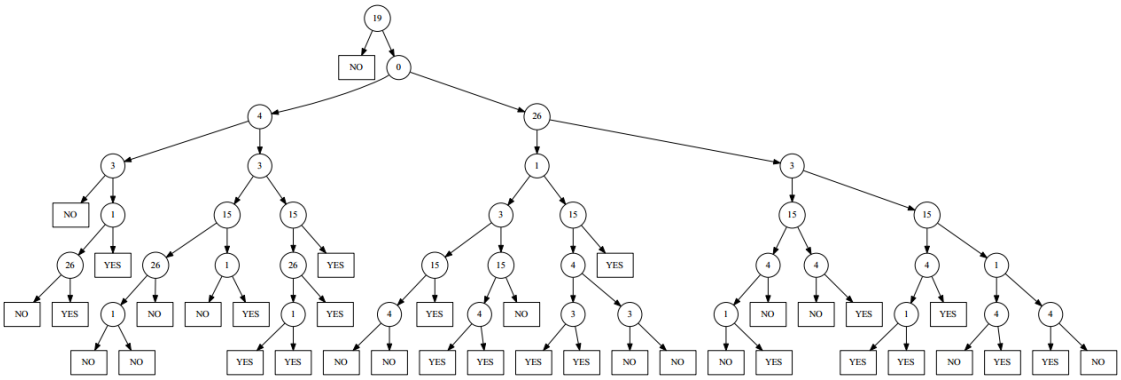

Figure 1.9: Decision tree with feature selection representing Fear

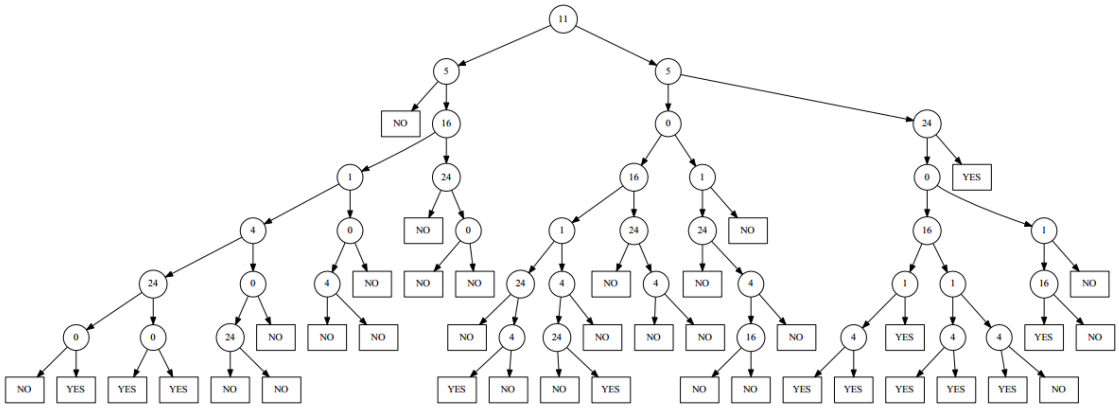

Figure 1.10: Decision tree with feature selection representing Happiness

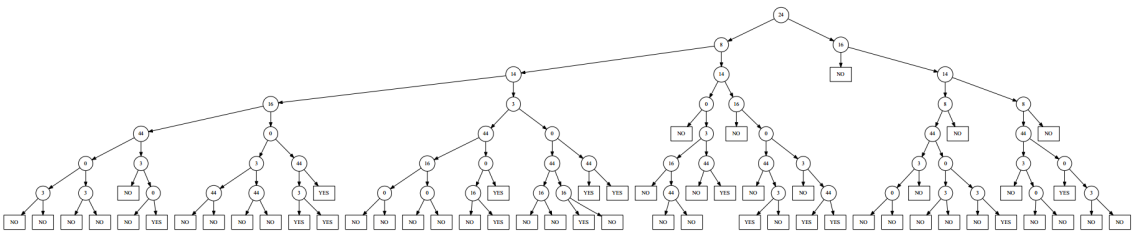

Figure 1.11: Decision tree with feature selection representing Sadness

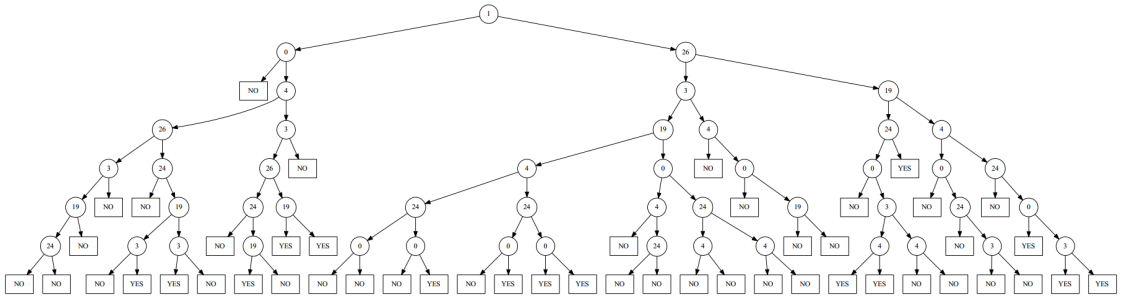

Figure 1.12: Decision tree with feature selection representing Surprise

1.2 System Evaluation

The cross validation is performed separately for clean dataset and noisy dataset. Similarly, the samples are split into 10 folds, 9 folds for training the data and 1 fold for testing. For each combination of folds, the set of 6 decision trees is constructed using training data. The set of trained decision trees will be used to make the prediction on the testing data. As a consequence, we will get 6 binary outputs, either positive or negative result for each emotion. These binary results will be integrated into single value, representing the class (1 to 6) of the emotion, and will be then compared with the actual results in order to estimate the error of the models. There will be cases such that there is more than 1 occurrence of positive output, or no occurrence of positive output (all negative outputs) received from set of 6 decision trees. Nevertheless, the solution to this issue will be further discussed in details on "Ambiguity Question" section.

The predicted and actual results, obtained from every combination of training and testing folds, are all used for building the confusion matrix and computing the average result of the recall, precision, F measure and classification rate in cases of clean dataset and noisy dataset.

Chapter 2

Evaluation

2.1 Confusion Matrix, and Performance Measurement

After performing cross validation for both clean dataset and noisy dataset, the confusion matrix and performance measurement could be achieved as shown in below sub-sections. The performance results illustrated here are from the implementation of (1) baseline decision tree with random selection, (2) baseline decision tree with depth selection, (3) feature selection with Wilson score interval, and (4) feature selection with Wilson score interval and voting system, respectively (see the detail on "Ambiguity Question" section).

2.1.1 Clean Dataset

Predicted / Actual	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**
Anger	85	11	5	5	19	3
Disgust	12	149	7	10	16	9
Fear	9	1	80	2	4	10
Happiness	7	11	4	187	11	9
Sadness	12	14	9	6	69	8
Surprise	7	12	14	6	13	168

Table 2.1: Confusion matrix (baseline decision tree with random selection) for clean dataset

Measurement, %	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**	**Average**
Recall Rate	66.4	73.4	75.5	81.7	58.5	76.4	72.0
Precision Rate	64.4	75.3	67.2	86.6	52.3	81.2	71.1
F1 Measure	65.4	74.3	71.1	84.0	55.2	78.7	71.6
Classification Rate	66.1	72.7	77.0	81.4	59.4	76.3	73.5

Table 2.2: Performance Measurement (baseline decision tree with random selection) for clean dataset

Actual \ Predicted	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**
Anger	91	8	5	7	17	5
Disgust	13	146	6	12	13	7
Fear	7	5	87	4	8	11
Happiness	5	12	2	177	9	10
Sadness	14	16	7	8	77	3
Surprise	2	11	12	8	8	171

Table 2.3: Confusion matrix (baseline decision tree with depth selection) for clean dataset

Measurement, %	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**	**Average**
Recall Rate	68.4	74.1	71.3	82.3	61.6	80.7	73.1
Precision Rate	68.9	73.7	73.1	81.9	58.3	82.6	73.1
F1 Measure	68.7	73.9	72.2	82.1	59.9	81.6	73.1
Classification Rate	69.0	73.5	72.8	82.5	61.5	81.3	74.6

Table 2.4: Performance Measurement (baseline decision tree with depth selection) for clean dataset

Actual \ Predicted	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**
Anger	94	6	3	0	13	0
Disgust	12	170	3	7	17	5
Fear	2	0	96	0	2	12
Happiness	3	6	3	204	2	4
Sadness	19	14	4	3	96	3
Surprise	2	2	10	2	2	183

Table 2.5: Confusion matrix (feature selection with Wilson score interval) for clean dataset

Measurement, %	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**	**Average**
Recall Rate	81.0	79.4	85.7	91.9	69.1	91.0	83.0
Precision Rate	71.2	85.9	80.7	94.4	72.7	88.4	82.2
F1 Measure	75.8	82.5	83.1	93.2	70.8	89.7	82.6
Classification Rate	78.9	79.4	85.5	92.4	68.8	91.5	84.0

Table 2.6: Performance Measurement (feature selection with Wilson score interval) for clean dataset

Actual \ Predicted	**Anger**	**Disgust**	**Fear**	**Happiness**	**Sadness**	**Surprise**
Anger	100	12	5	0	10	0
Disgust	7	165	4	2	15	5
Fear	3	0	95	1	2	8
Happiness	4	8	1	206	3	2
Sadness	17	12	2	2	95	5
Surprise	1	1	12	5	7	187

Table 2.7: Confusion matrix (feature selection with Wilson score interval and voting system) for clean dataset

Measurement, %	Anger	Disgust	Fear	Happiness	Sadness	Surprise	Average
Recall Rate	78.7	83.3	87.2	92.0	71.4	87.8	83.4
Precision Rate	75.8	83.3	79.8	95.4	72.0	90.3	82.8
F1 Measure	77.2	83.3	83.3	93.6	71.7	89.0	83.1
Classification Rate	79.8	82.9	87.1	91.5	72.5	88.1	84.5

Table 2.8: Performance Measurement (feature selection with Wilson score interval and voting system) for clean dataset

2.1.2 Noisy Dataset

Actual \ Predicted	Anger	Disgust	Fear	Happiness	Sadness	Surprise
Anger	27	9	19	14	22	12
Disgust	13	120	20	13	13	9
Fear	20	16	100	11	11	13
Happiness	10	18	25	152	14	16
Sadness	14	14	12	10	42	14
Surprise	4	10	11	9	8	156

Table 2.9: Confusion matrix (baseline decision tree with random selection) for noisy dataset

Measurement, %	Anger	Disgust	Fear	Happiness	Sadness	Surprise	Average
Recall Rate	26.2	63.8	58.5	64.7	39.6	78.8	55.3
Precision Rate	30.7	64.2	53.5	72.7	38.2	70.9	55.0
F1 Measure	28.3	64.0	55.9	68.5	38.9	74.6	55.1
Classification Rate	28.3	64.4	57.2	64.6	39.6	79.7	59.6

Table 2.10: Performance Measurement (baseline decision tree with random selection) for noisy dataset

Actual \ Predicted	Anger	Disgust	Fear	Happiness	Sadness	Surprise
Anger	20	12	10	9	24	9
Disgust	19	122	15	12	12	11
Fear	22	13	112	10	12	14
Happiness	11	18	17	158	10	11
Sadness	7	12	11	9	40	10
Surprise	9	10	22	11	12	165

Table 2.11: Confusion matrix (baseline decision tree with depth selection) for noisy dataset

Measurement, %	Anger	Disgust	Fear	Happiness	Sadness	Surprise	Average
Recall Rate	23.8	63.9	61.2	70.2	44.9	72.1	56.0
Precision Rate	22.7	65.2	59.9	75.6	36.4	75.0	55.8
F1 Measure	23.3	64.6	60.5	72.8	40.2	73.5	55.9
Classification Rate	24.4	64.4	61.6	71.1	44.6	71.1	61.6

Table 2.12: Performance Measurement (baseline decision tree with depth selection) for noisy dataset

Actual \ Predicted	Anger	Disgust	Fear	Happiness	Sadness	Surprise
Anger	36	6	12	7	13	3
Disgust	13	158	10	11	11	5
Fear	17	10	140	14	20	18
Happiness	5	4	9	167	4	6
Sadness	12	4	6	2	57	8
Surprise	5	5	10	8	5	180

Table 2.13: Confusion matrix (feature selection with Wilson score interval) for noisy dataset

Measurement, %	Anger	Disgust	Fear	Happiness	Sadness	Surprise	Average
Recall Rate	46.8	76.0	63.9	85.6	64.0	84.5	70.1
Precision Rate	40.9	84.5	74.9	79.9	51.8	81.8	69.0
F1 Measure	43.6	80.0	69.0	82.7	57.3	83.1	69.5
Classification Rate	46.8	75.1	63.1	85.6	63.5	84.6	73.7

Table 2.14: Performance Measurement (feature selection with Wilson score interval) for noisy dataset

Actual \ Predicted	Anger	Disgust	Fear	Happiness	Sadness	Surprise
Anger	39	8	11	9	16	2
Disgust	14	158	11	9	8	3
Fear	17	13	139	13	13	12
Happiness	4	3	10	168	3	9
Sadness	11	3	8	2	62	5
Surprise	3	2	8	8	8	189

Table 2.15: Confusion matrix (feature selection with Wilson score interval and voting system) for noisy dataset

Measurement, %	Anger	Disgust	Fear	Happiness	Sadness	Surprise	Average
Recall Rate	45.9	77.8	67.1	85.3	68.1	86.7	71.8
Precision Rate	44.3	84.5	74.3	80.4	56.4	85.9	71.0
F1 Measure	45.1	81.0	70.6	82.8	61.7	86.3	71.4
Classification Rate	48.0	77.7	66.8	85.2	65.5	86.7	75.4

Table 2.16: Performance Measurement (feature selection with Wilson score interval and voting system) for noisy dataset

2.2 Analysis of the Cross Validation Experiments

Intuitively, cross-validation is used for evaluating the prediction performance of our model and how well it predicts the result on unseen samples. The intuition of using this technique is that

when we fit a model with our data, we expect it to well perform for not only in-sample data, but for out-sample data as well. Hence, from the feedback we acquire from out-sample data, we could figure out the real performance of our model under new dataset and see whether there are any needs of improvement. Without cross-validation, we only have the information about the performance of our model on our training data, but no clue about the model's actual prediction capability. In other words, when the model performs well on training set, we could not really decide whether it captures the intrinsic relationship of the model or it just overfits the data. In addition, 10-fold cross-validation is robust in evaluating our performance on out-sample data in case there are some random biased issues in one of the 10 experiments of 10-fold cross-validation, such that we can get the reliable performance of the trained model.

By applying Wilson Score Interval and Voting System in handling the ambiguity, the performance measurement (recall rate, precision rate, F1 measure, and classification rate) in average improves by roughly 10% for clean dataset, 15% for noisy dataset. The large gap between before and after implementation of these 2 approaches implies that in general, the decision trees have some difficulties in distinguishing different emotions, leading to occasional occurrences of multiple positive output or all negative outputs. Thus, without proper methodology to deal with this vagueness, the original trees will likely produce faulty result.

In general, the recall rate, precision rate, F1 measure and classification Rate are close. In this case, we regard the dataset as balanced dataset. When taking a closer look at the result obtained from implementing both Wilson Score Interval and Voting System, the prediction accuracy is considerably high by overall. For clean dataset, the recall rate, the precision rate, F1 measure, and classification rate are 83.4%, 82.8%, 83.1%, and 84.5%. For noisy dataset, those measurements reduce to 71.8%, 71.0%, 71.4%, and 75.4%, respectively (approximately 10% deduction for every measurement). The prediction accuracy is highly achieved in 3 emotions, consisting of Disgust, Happiness, and Surprise. The classification rate of those 3 emotions are 82.9%, 91.5%, and 88.1% in case of clean dataset, and 77.7%, 85.1%, and 86.7% in case of noisy dataset. The reduction from inducing noisy data is relatively small for these 3 emotions. For Anger, the classification rate drastically decreases from 79.8% to 48.0% (approximately 30% difference), indicating that the prediction performance is highly sensitive to the noise in the data. The reason behind this phenomena might be that the attributes of Anger largely overlap with other emotions, and therefore could be easily disturbed by those noisy data. Furthermore, from the observation of confusion matrix tables, we can see that the anger emotion tends to be classified into fear, disgust into sadness and vice versa. From the perspective of human, these emotions have certain similarities to some extent. Thus, it is reasonable for the model to mis-classify them.

Chapter 3

Questions

3.1 Noisy-Clean Datasets Question

In the absence of any pruning techniques and completely clean dataset, the ID3 algorithm attempts to capture or memorise every single instance, instead of generalising them. Indeed, the introduction of noisy data into training dataset could adversely affect the tree construction by taking erroneous information into an account, leading to some unnecessary branching. By inducing pre-feature selection using random forest, some irrelevant attributes from noisy dataset will be excluded in prior to tree training, which helps avoiding overfitting data.

Furthermore, since noise disturbs the construction of tree, it will absolutely reduce the effectiveness of prediction (15% deduction of classification rate between clean and noisy is observed when applying random selection and depth selection), and hence likely aggravate ambiguity problem (for example, increase the number of multiple positive outputs). By implementing Wilson Score Interval and Voting System, this problem is alleviated with roughly 15% improvement of classification rate in case of noisy data. Plus, the difference between classification rate of clean and that of noisy dataset has reduced from 15% to 10%. However, the problem is still severe in case of Anger emotion where approximately 30% difference of classification rate between clean dataset and noisy dataset is observed. It is likely that when the attributes of one specific emotion are overlapped to that of other emotions, the noisy data could easily confuse the algorithm. Meanwhile, the gap between classification rate of Disgust, Happiness, and Surprise is around 5%, indicating that the attributes of these emotions are so distinct such that introducing small amount of noise could not affect the majority data when training.

3.2 Ambiguity Question

If we solely input the sample into 6 individual decision trees without combining the results, there is possibility of achieving multiple positive outputs, or no positive output at all. Initially, we use the random selection, which means we randomly select the label from those yielding positive outputs or from 1 to 6 in case of all negative outputs. Nevertheless, this approach is way naive; without any knowledge to support, it could easily lead to false prediction. For instance, if there are 2 positive outputs, there will be only 50% that we could achieve the correct label. To resolve this issue, 3 approaches have been considered. In the first approach, namely depth selection, we will keep tracking the depth of path in concurrence with evaluating the binary output for each tree. If there is an existence of emotions giving positive results, one with shortest path - or in other word, utilising fewer number of attributes in evaluation - will be selected in order to avoid using too specific hypothesis. Nevertheless, if it turns out to be more than 1 emotion matching that criteria, the label will be randomly chosen from one of those. In the case that all decision trees return all negative outputs, we simply random the label from 1 to 6. The advantage of this approach is its simplicity. Nevertheless, the assumption to choose shortest branch sometimes could lead to too general hypothesis as well.

For the second approach, we use "Wilson Score Interval" to help selecting the class of emotion based on its confidence (for more details, please see [3]). Basically, we might end up using

'majority case' as a leaf node when we train a tree. From this reason, the decision trees might possibly give the same output, but indeed occupied with different level of confidence. However, we could not simply calculate the confidence because there is a case where the multiple output yield the same value of confidence but supported with different number of samples. For example, let say that there are 10 positive targets and 40 negative targets. In this case, we get a negative leaf node with 80% confidence. Nevertheless, it should be note that this 80% confidence is calculated only from 50 data points. Obviously, if we have 20 positive targets and 80 negative target, we also obtain negative leaf node with 80% confidence, but in this case, we actually use more data to gain our confidence, such that this [80% confidence for 20 positive targets and 80 negative targets] should be higher compared to [80% confidence for 10 positive targets and 40 negative targets] in term of confidence. Thus, we decide to use Wilson Score Interval as our alternative for this issue.

$$\frac{n_S+\frac{z^2}{2}}{n+z^2} - \frac{z}{n+z^2}\sqrt{\frac{n_S n_F}{n}+\frac{z^2}{4}}$$

Where n_S is the number of our majority case (in the previous example, 40 or 80), z is the confidence factor, here we select z = 1.96, n is the number of all targets, n_F is the number of minority case (in the previous example, 10 or 20)

Finally, we have applied the "Voting System". To build the voting system, we first train extra 15 trees specially for voting including 1Versus2, 1Versus3, 1Versus4, 1Versus5, 1Versus6, 2Versus3, 2Versus4, 2Versus5, 2Versus6, 3Versus4, 3Versus5, 3Versus6, 4Versus5, 4Versus6, 5Versus6. 1Versus2 denotes a tree trained to classify emotion 1 and emotion 2, regarding emotion 1 as negative (0), and emotion 2 as positive (1). As a result of possible combination, the number of our extra trees is (6 2) = 15.

Then, when our base 6 decision trees return multiple positive results or all negative results (in all negative case, we will have to 6 choices to select), we take this example to do the evaluation with our 15 extra trees. For example, the first tree (1Versus2) may tell us that this example is that of emotion 1, and the second tree (1Versus3) may tell us this example is that of emotion 1 again, and so on. As a consequence, we get 15 results in total from 15 tress, and we simply decide the end result to be the emotion with the highest vote from 15 trees.

Unlike the first approach which involves purely intuitive assumption about the depth, Wilson Score Interval and Voting System attempt to distinguish the emotion by actual information, and thus could effectively improve overall prediction performance: 10% for clean dataset and 15% for noisy dataset. Indeed, both 2 approaches occupy the binary output with some confidence measures and use it to justify when ambiguity arrives. Unfortunately, the downside of these 2 approaches is obviously the complexity, especially in case of the Voting System which requires the construction of additional 15 trees.

3.3 Pruning Question

Firstly, pruning_example invokes **classregtree** function which creates a decision tree T for predicting response Y as a function of predictors X without pruning. Then, it invokes **test** function to compute the cost, the standard error of cost value, the number of terminal nodes for each tree, and the estimated best level of pruning using cross-validation method and re-substitution method, respectively. The **prune** function generates pruned trees based on the best level of pruning obtained above. The **min** function is invoked to get minimum cost and its index for both methods. Finally, pruning_example plots the curves presenting the relationship of cost and tree size.

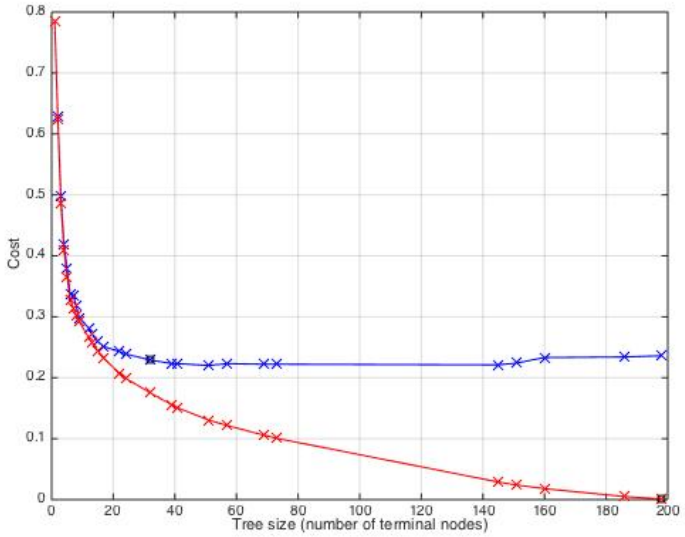

Figure 3.1: Plot from prunning_example function from clean dataset

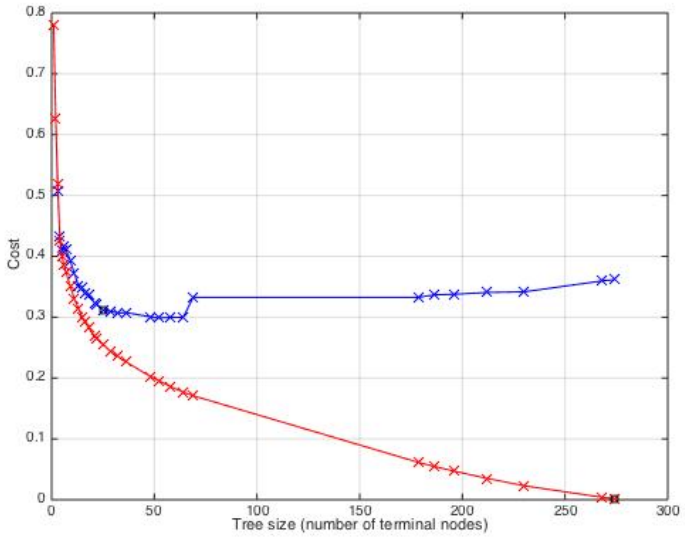

Figure 3.2: Plot from prunning_example function from noisy dataset

Above figures illustrate the result plot from pruning_example function for clean dataset and noisy dataset respectively. The red curve represents the cost computed using the re-substitution method while the blue curve represents the cost computed using 10-fold cross-validation method. The re-substitution cost is based on the same sample that was used to create the original tree, so it is virtually the training error and underestimates the likely cost of applying the tree to new data. Thus, when the tree size grows, it keeps decreasing until it reaches 0, indicating that the model the data. For the cross-validation method, the samples are first partitioned into 10 subsamples randomly. For each subsample, **test** function fits a tree to the remaining data, uses it to predict the subsample, and computes the cost for the subsample. The cost computed using this method could be regarded as validation error since it is based on the sample different from the training sample. Thus, when the tree size grows, cross-validation cost decreases first but remains flat later, indicating that the model tends to overfit the training dataset. In this case, the model has a high performance on training set but cannot extend the usage to validation set, inducing the huge gap of cost between training and validation set. Compared with that of clean dataset, the cost of

validation curve of noisy dataset suddenly increases when the tree size increases from 64 to 69. This phenomenon is possibly a result of the noise in the dataset. When the tree size becomes large, the noise tends to disturb and introduce an irrelevant information to the construction of the tree, which reduces the accuracy of the model.

In conclusion, the pruning technique conveys the appropriate level of generality, preventing the decision trees from being too general or too specific. From the observation of the experimental results, the optimal tree size for clean and noisy dataset are 32 nodes and 25 nodes respectively.