CO395 Coursework 1 Report

Authors: Peizhen Wu, Jiajun Chen, Kai Zhang, Zhen Hao

1.**Brief summary of implementation details.**

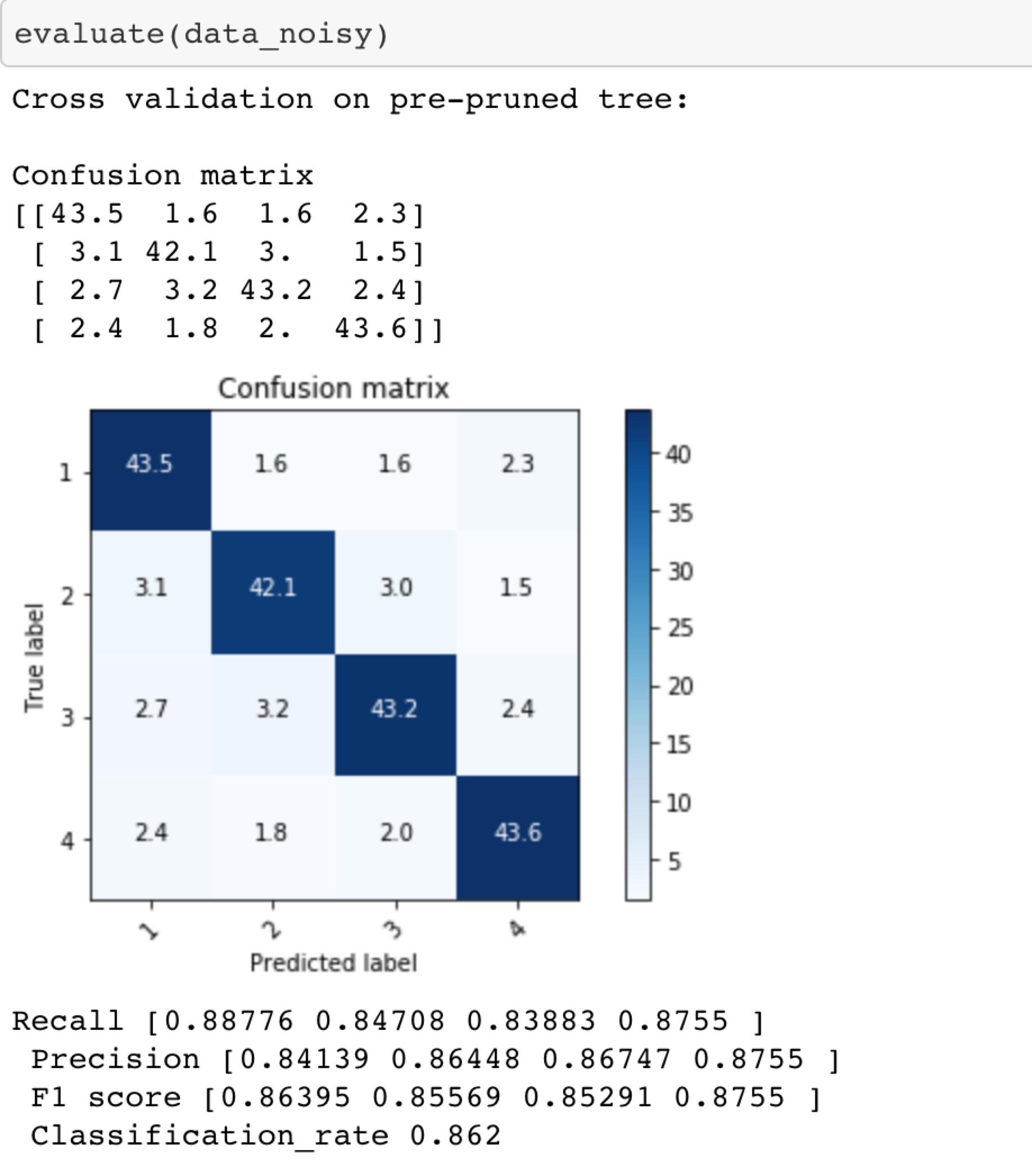
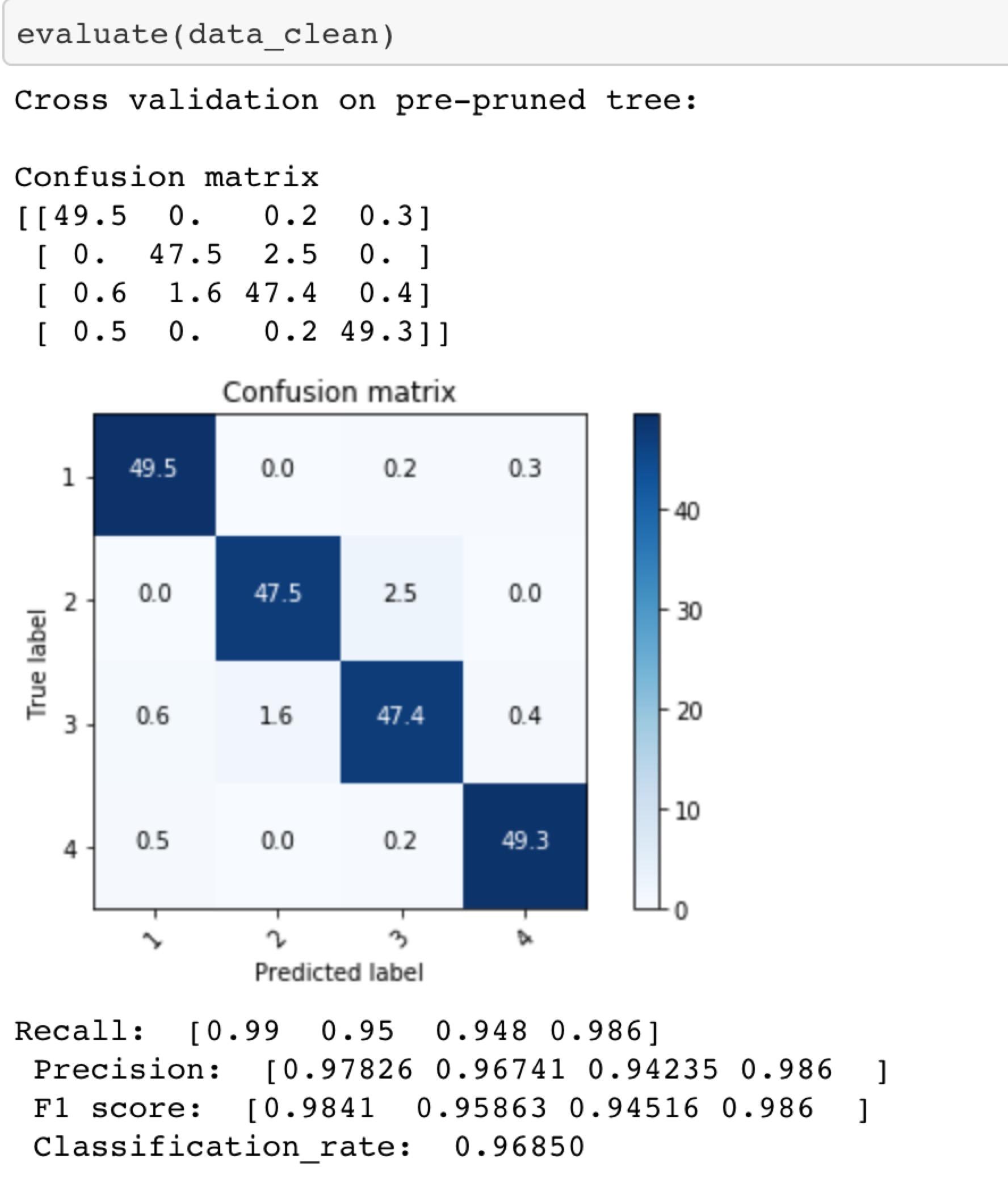
We create a tree class object to represent a node in the decision tree, each node has attributes ‘split attribute’, ‘value’, ‘left’ and ‘right’. It call buildTree() function recursively to create new nodes on ‘left’ branch and ‘right’ branch.The split attribute and value for new node is determined by ‘find\_split() function. We select the best split attribute and value greedily by checking each value on each attribute as a candidate split, then find out the one that has the highest information gain. Once the split point is determined, we assign the sub-dataset to the child nodes which will later be used for pruning. A leaf node is defined when all labels in the sub-dataset are the same and its class is the label in the sub-dataset. There is a case that the algorithm choose a split in which all rows belong to one group(i.e. No information gain). In this case, we terminate the process by defining it as a leaf node and assign its class by majority vote in sub-dataset label. With this method, we just need to create the root node and pass the train dataset to it, the whole tree will be built recursively.

Making prediction involves navigating the tree by checking if the split attribute and value is satisfied. The procedure will repeat until it reaches a leaf node of the tree and the prediction is made by the class value of the leaf.

To evaluate and qualify the performance of the decision tree we have built, we perform 10-fold cross validation on the given dataset. We partition the index of the data into 10 balanced arrays, each time we choose one array of indices of data as a validation set, and build a tree using the remaining 9 folds of indices of data. We construct and record the confusion matrix for each validation set and find out the average over 10 different models. The average classification rate as well as other metrics like recall and precision rate can then be calculated. We have similar implementation for the prune cross validation, except that we split the dataset into 10 folds and train the tree using 8 of them, then prune the tree using one of the remained folds (validation set) and measure the record the accuracy as well as confusion matrix based on the last fold (test set).

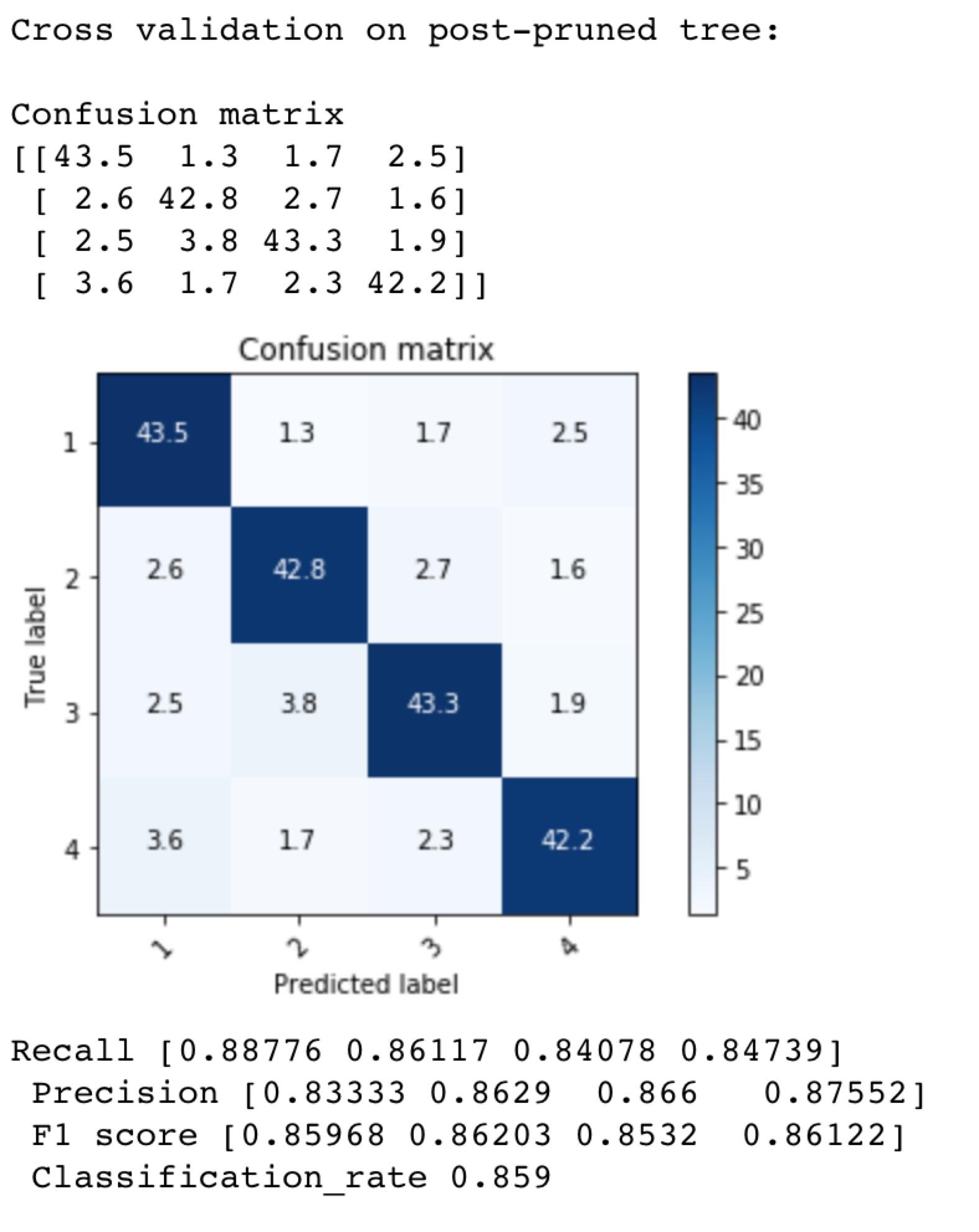
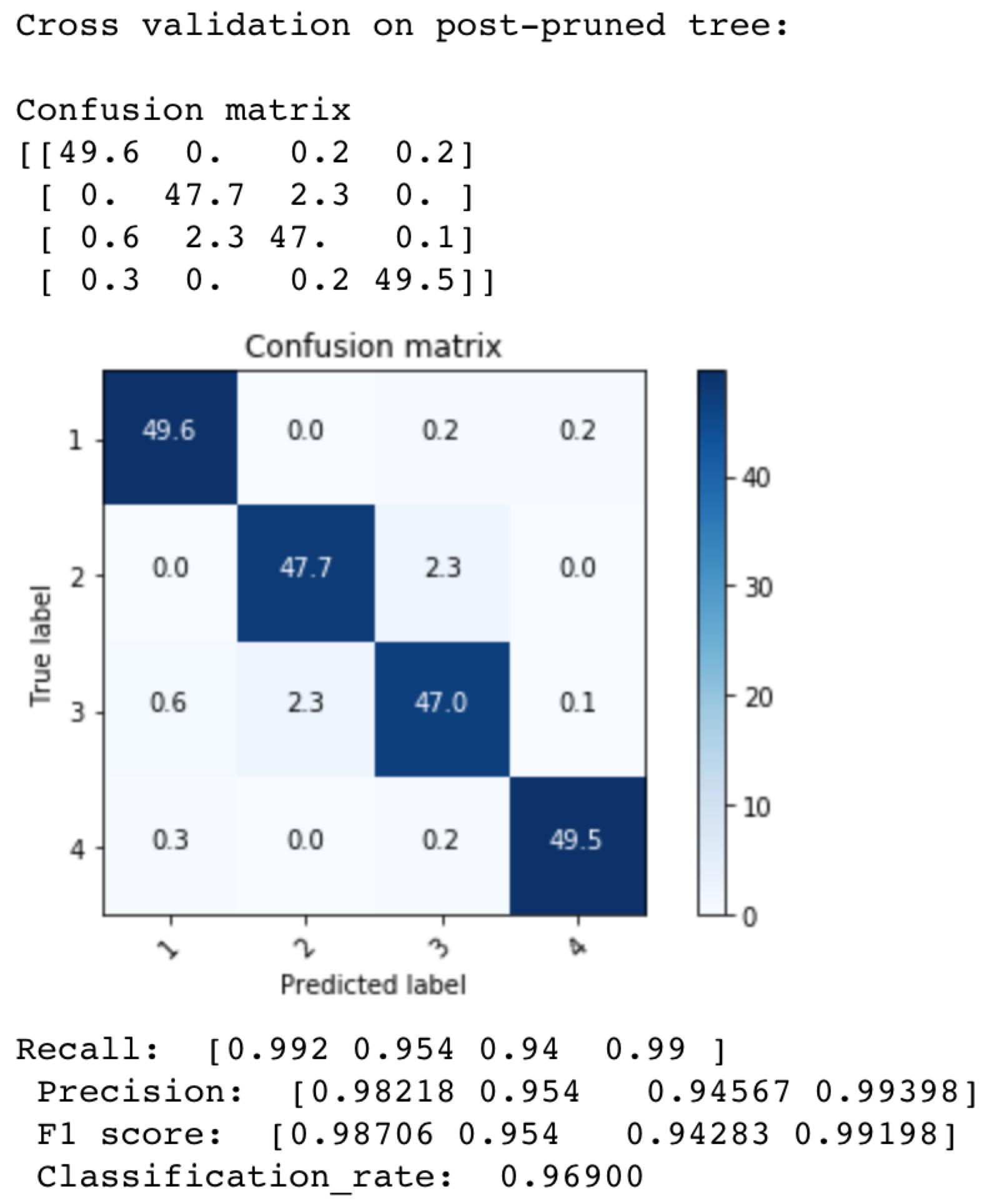
We train and evaluate two decision trees for clean and noisy dataset separately, the results will be discussed later.

**2.Commented 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 four classes; for both clean and noisy datasets.**



**Clean Dataset Noisy Dataset**

* The confusion matrix is the average result of 10 cross-validations. By observing the confusion matrix calculated based on the clean dataset, we can find that class 1 and class 4 are more likely to be classified correctly whereas class 2 and class 3 are likely to be misclassified as each other. We can also observe this pattern in the average recalls, the average precision and F1 scores.
* The classification rate represents the accuracy of data classified by the decision tree. The result, 0.9685 means 96.85% of the total data can be classified correctly by the trained decision tree.
* The average recall represents the percentage of the selected items out of the total relevant items, which is the coverage of the data from class 1-4. For example, 99% data from actual class 1 are correctly identified, 95% data from actual class 2 are correctly identified, 94.8% data from actual class 3 are correctly identified, and 98.6% data from actual class 4 are correctly identified. In general, most of the data from the four classes are covered in the decision tree.
* The average precision represents the percentage of the relevant items out of the selected items, which is the accuracy of the predicted data. For class 1, about 97.83% of the predicted data are correct. For class 2, about 96.74% of the predicted data are correct. For class 3, about 94.24% of the predicted data are correct. For class 4, about 98.6% of the predicted data are correct. In general, most of the classifications are correct in the decision tree.
* In this coursework, F1 score is the harmonic mean of precision and recall. It considers precision and recall equally. By observing the result, we can find that class 4 can be classified accurately the most, followed by class 1, class 2, and class 3.
* Comparing the confusion matrix calculated based on the noisy dataset with the clean dataset, we can find that data are more likely misclassified. The predicted data covers less actual data and they are less precise.
* The classification rate simply shows the accuracy of this decision tree, which can only classify 86.2% of total data.
* The average recall from the noisy confusion matrix shows that less data from one specific class are identified. The predictions of class 1 covers the most data from actual class 1, about 88.78% data of class 1 are identified.
* Similarly, the average precision from the noisy confusion matrix shows that less predicted data are correct. The predicted class 4 is most precise, 87.55% predicted data are correct.
* F1 score also shows the noisy decision tree identifies less data and has a higher probability to make mistakes.

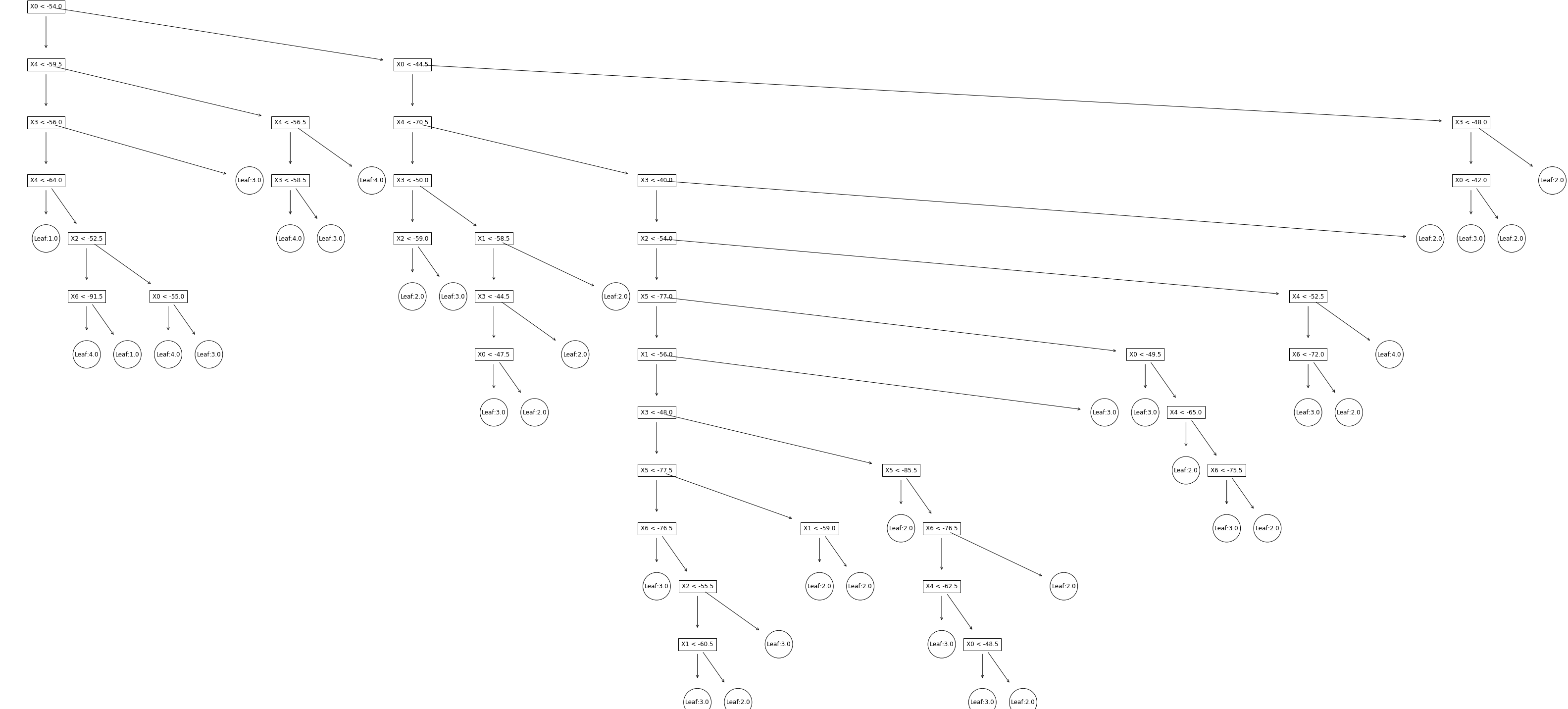


The left picture is the evaluation of the pruned clean tree. The right picture is the evaluation of the pruned noisy tree. Observing the results after pruning, we can find the classification rate of the clean tree slights increases, whereas the classification rate of the noisy tree decreases. The result is reasonable because the datasets for pruning and testing are different. Pruning can guarantee that predicting on the prune set outputs a better result rather than on the test set. Also, when we build the decision tree, the depth of the tree is not deep enough, pruning may result in decreasing the classification rate of both clean and noisy decision trees.

The evaluation results may be different when re-running the code because we randomly shuffle the dataset at the beginning, so the decision tree can be trained differently by using different dataset. The confusion matrix and other evaluation metrics will change according to different decision trees.

**3.Diagrams of the trees trained on the entire dataset (bonus points)**

The plot can be found on the file *treeplot.png*



**4.Answers to the three questions of Part 2.**

1)**Noisy-Clean Datasets Question Is there any difference in the performance**

**when using the clean and noisy datasets? If yes/no explain why. Discuss the**

**differences in the overall performance and per class.**

Yes. The classification rate of the decision tree when using the noisy dataset is significantly lower than using the clean dataset. Noise made it difficult for decision tree to find the relationship between the features and labels. Decision tree tried to fit the noise and so its generalization ability was reduced. Observing model’s performance on the four classes on the noise dataset, it can be inferred that the noise is approximately evenly distributed into four categories, because the degree of performance degradation for each class is similar.

2)**Pruning Question Briey explain how you implemented your pruning func-**

**tion and details the influence of pruning on the accuracy of your tree on**

**both datasets**.

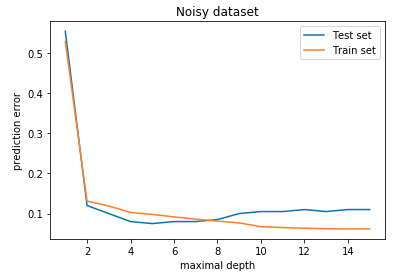
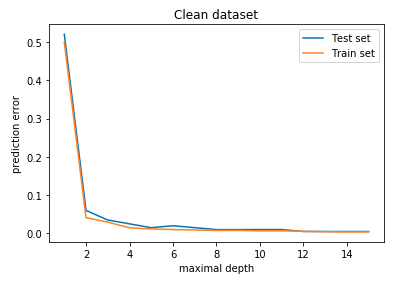
The pruning method of class Decision Tree is designed as a recursive procedure, the basic idea is to take the validation date as an input to a trained tree(could be a subtree). In the case that the two nodes of this tree are both leaf nodes,our algorithm trys to find that whether deleting both leaf nodes can improve the validation accuracy of this tree by simply counting the number of correctly classified data points in both circumstance, if so, it changes the input tree to a node with a class attribute given by majority vote.In case that the input tree has one or more nodes that are not leaf nodes, our algorithm recursively calls itself on every subtree.

We saw a slight decrease of accuracy on our clean dataset after pruning our decision tree. The reason could be that the validation set and the test set are different, which leads to some inconsistency. On the noisy dataset ,the increase is especially significant, We assume the reason to be that before pruning the decision tree, noises are taken into account in forming the tree’s structure. This negative effect is mostly evident at leaf nodes level. The pruning mechanism can prevent the tree from fitting to the noises in the input to some degree, which will lead to an improved accuracy on test set.

3)**Depth Question** **Comment on the maximal depth of the tree that you gen-**

**erated for both datasets and before and after pruning. What can you tell about**

**the relationship between maximal depth and prediction accuracy?**



For this part, we divide the data set into a training set and a test set instead of using k-fold. For clean datasets, the decision tree model can achieve its best performance with maximal depth equal to or greater than 10. For clean datasets, the decision tree model can achieve its best performance with maximal depth 8, increasing the depth would reduce the model’s performance on test set. After prune, the optimal maximal depth for both datasets did not change. Trimming out the excess leaves improves the performance of the model on test set, but prune does not necessarily reduce the maximum depth of the tree because some leaves that reach the maximum depth are indeed useful.

**Relationship between maximal depth and prediction accuracy:**

The deeper the tree, the more complex the model and the higher the degree of freedom.

For training error, for both datasets, it is easy to see that as the maximum depth is increased, the training error will always decrease, or at least not increase.

For the testing error, it is not so obvious. If the maximum depth is set too high, then the decision tree will overfit the training data without capturing the useful patterns we want, which can cause the test error to rise. This is obvious on the noisy data set. But if the maximum depth is set too low, it is not very good as well, because the decision tree is too restricted, so that it can not capture the pattern, which will lead to increased test error. As we have seen, the test error of both data sets is high when the maximum depth is small. There is a best trade-off point between the highest and the lowest.