Homework 4

*Note: This assignment uses Python3.

1. At the very beginning of the report, summary of the overall accuracy on the test dataset for each dataset (2 methods * 4 datasets = 8 numbers). These metrics are used to score the performance of the model, which is to be detailed in the next two sections;

	Decision Tree	Random Forest
balance.scale	73.5%	73.1%
nursery	96.3%	96.2%
led	86.1%	86.1%
synthetic.social	48.1%	56.6%

***Since the number of attributes is small, the number of attributes for random forest is same as decision tree turns out to be the best choice. Thus, the result seems similar.

2. Brief introduction of the classification methods in your classification framework;

(1) Decision Tree: In this assignment, I use decision tree with gini-index, considering every possible categorical value of the feature as a branch. I traverse each value of each categorical to calculate gini-index, then choose the best value of current node's value/ the best next splitting point as condition to class dataset. The label of the leaf will be used majority vote to determine the class of the leaf. Finally, recursively build up the tree and then data can be classified to the leaf of the tree.
(2) Random Tree: Loop and build up several decision trees as above, choosing subset of attributes. Then use vote to determine the label of the data.

3. All model evaluation measures you calculated above ((1 overall metric + # class * 1 per-class metric) * (1 on training set + 1 on test set) * 2 methods * 4 datasets);

(1) balance.scale.train:

- Decision Tree:

- Accuracy: 93.5%

- F-1 Score:

Class	F-1 Score
1	0.455
2	0.950
3	0.946

- Random Forest:

- **Accuracy:** 93.5%
- **F-1 Score:**

Class	F-1 Score
1	0.455
2	0.950
3	0.947

(2) balance.scale.test:

- Decision Tree:

- **Accuracy:** 73.5%
- **F-1 Score:**

Class	F-1 Score
1	0.0
2	0.811
3	0.770

- Random Forest:

- **Accuracy:** 73.1%
- **F-1 Score:**

Class	F-1 Score
1	0.0
2	0.811
3	0.770

(3) nursery.train:

- Decision Tree:

- **Accuracy:** 98.3%%
- **F-1 Score:**

Class	F-1 Score
1	0.973
2	0.895
3	0.975
4	0.997
5	0.25

- Random Forest:

- **Accuracy:** 99.1%
- **F-1 Score:**

Class	F-1 Score
1	0.985
2	0.932
3	0.986
4	0.997
5	0.352

(4) nursery.test:

- Decision Tree:

- Accuracy: 96.6%

- F-1 Score:

Class	**F-1 Score**
1	0.947
2	0.818
3	0.957
4	0.996
5	0.0

- Random Forest:

- Accuracy: 96.2%

- F-1 Score:

Class	**F-1 Score**
1	0.914
2	0.603
3	0.928
4	0.996
5	0.0

(5) led.train:

- Decision Tree:

- Accuracy: 85.9%

- F-1 Score:

Class	**F-1 Score**
1	0.767
2	0.898

- Random Forest:

- Accuracy: 84.8%

- F-1 Score:

Class	**F-1 Score**
1	0.722
2	0.895

(6) led.test:

- Decision Tree:

- Accuracy: 86.2%

- F-1 Score:

Class	**F-1 Score**
1	0.774
2	0.900

- Random Forest:

- **Accuracy:** 86.3%
- **F-1 Score:**

Class	F-1 Score
1	0.773
2	0.902

(7) synthetic.social.train:

- Decision Tree:

- **Accuracy:** 90.0%
- **F-1 Score:**

Class	F-1 Score
1	0.890
2	0.891
3	0.903
4	0.900

- Random Forest:

- **Accuracy:** 99.0%
- **F-1 Score:**

Class	F-1 Score
1	0.985
2	0.984
3	0.984
4	0.983

(8) synthetic.social.test:

- Decision Tree:

- **Accuracy:** 48.6%
- **F-1 Score:**

Class	F-1 Score
1	0.475
2	0.440
3	0.492
4	0.513

- Random Forest:

- **Accuracy:** 59.5%
- **F-1 Score:**

Class	F-1 Score
0	0.618
1	0.543
3	0.591
4	0.594

4. Parameters you chose during implementation and why you chose these parameters;

(1) balance.scale:

- Decision Tree:

- threshold: 0.1 (When threshold = 0.1, it turns out to be the best choice. Since category is small and each category doesn't have a lot of values, threshold should be small)

- Random Forest:

- threshold: 0.1

- idx_num(number of attributes): 4

- iter = 20

(Since the number of attributes is small, the number of attributes for random forest is same as decision tree turns out to be the best choice. Chose number of attributes same as the original one and small iter times is enough. Thus, the result seems similar.)

(2) nursery:

- Decision Tree:

- threshold: 0.0

(When threshold = 0.0, it turns out to be the best choice. Since category is small and each category doesn't have a lot of values, threshold should be small)

- Random Forest:

- threshold: 0.1

- idx_num(number of attributes): 7

- iter = 20

(Since the number of attributes is small, the number of attributes for random forest is 7 as decision tree turns out to be the best choice. Chose number of attributes same as the original one and small iter times is enough. Thus, the result seems similar.)

(3) led:

- Decision Tree:

- threshold: 0.2

(When threshold = 0.2, it turns out to be the best choice. Since category is small and each category doesn't have a lot of values, threshold should be small)

- Random Forest:

- threshold: 0.2

- idx_num(number of attributes): 6

- iter = 20

(Since the number of attributes is small, the number of attributes for random forest is 6 as decision tree turns out to be the best choice. Chose number of attributes same as the original one and small iter times is enough. Thus, the result seems similar.)

(4) synthetic.social:

- Decision Tree:

- threshold: 0.0

(When threshold = 0.0 it turns out to be the best choice.)

- Random Forest:

- threshold: 0.0

- idx_num(number of attributes): 100

- iter = 10

(Since the number of attributes is large, choose the number of attributes for random forest is smaller as decision tree to improve speed.)

4. Your conclusion on whether the ensemble method improves the performance of the basic classification method you chose, why or why not.

As showed above, for dataset 1-3, which number of attributes are small. Ensemble method does not improve or improve litter the performance of the basic classification method. Since the number of attributes are small, the deep of train is small, it's possible to traverse each attribute to get the exact condition value for each node.

While for the last large dataset, ensemble method improves the performance a lot, since it's time consuming to traverse each attribute every time. Using ensemble method let us choose several subsets of attribute and train them using multi-thread, which is a good way to balance time and performance. With majority vote, the performance turns out better.