Assignment 4 Report
Lingsong Gao, lg5

1. Classification Methods:

1) Basic method: gini-index

Gini-index is used as the basic classification method to select the best attribute for each node in the decision tree which has smallest information impurity /gini index. In this assignment, all the features are categorical with multi-values. Therefore, we use full split to split every node into N branch where N is the number of possible values of the feature. And the gini index is calculated as:

$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$, p_j is the relative frequency of class j in D

$$gini_A(D) = \sum_{i=1}^{k} \frac{|D_i|}{|D|} gini(D_i)$$

k is the number of possible values of Attribute A, and Di is the number of tuples in D where attribute A has value i.

Using the above expressions to calculate gini-index of every attribute in candidate attributes of every node, we can choose the best attribute with smallest gini-index to fully split the node according to all possible values of this attribute. We continue the splitting with four kinds of terminal conditions:

i) The tree height reaches max_depth.
ii) The number of data is below the minimum number for each node.
iii) Label of all the data in the node is the same and there is no point to split.
iv) All attributes have been selected and no more attribute remains.

2) Ensemble method: random forest

Using gini-index as the basic method to build every individual decision tree, I then used both data bagging and feature bagging to continuously build trees and ensemble all generated trees to make classifications.

I used sample with replacement method for data bagging to sample training data for each tree. And for the tree node of every tree, I used feature bagging to randomly select F attributes in the remaining attribute list as candidates and used gini-index to select the best attribute to split the tree.

In the end, the label assigned to unseen dataset is made by the majority vote of all constructed decision trees.

2. Parameters Selection:

1) Decision Tree:

Min_num(minimum number of data points required for a node to keep splitting) is set to 1 to stop splitting when there remains only one data point.

Thus, there is only one parameter that needs to be tuned for different dataset: max_depth. We need to choose an appropriate max_depth to ensure accuracy but not too large to cause overfitting.

2) Random Forest:

There are 5 parameters in random forest model: max_depth, min_num, sample_ratio, tree_number and F_features.
Max_depth is the maximum depth of each decision tree and since every tree can grow to its largerst size in random forest method, we don't set limits to max_depth and thus the max_depth is set to number of features in training set.
Min_num is set to 1 as in decision tree to stop splitting.
Sample_ratio is the parameter used to control data bagging and is selected as 1 to consider all training set for each decision tree.
Thus, there are only 2 features that need to be tuned in random forest model: tree_number and F_features.

In general, larger tree_number can bring about better results but can also increase running time and may not enhance performance above certain threshold.
F_features is the number of candidate features to be selected for every node using gini-index. According to research, F_features is better determined as square root of number of features, and I did some adjustments based on this empirical result to enhance performance.

3) Parameters for each dataset:
i) balance.scale:
a. Decision Tree: max_depth = 3
There are four features in this dataset and the performance based on different max_depth are:
1: 0.551, 2: 0.609, 3: 0.613, 4: 0.613 (overall accuracy)
Max_depth above 3 to its maximum 4 cannot improve performance so we can just set max_depth = 3.
b. Random Forest: F_features = 1, tree_number = 500
I tried 1-4 F_features number but only find that F_features = 1 can yield best result. I think maybe there are only 4 features so that randomly select one is the best choice. And for tree_number, 500 trees can ensure good performance while the running time is still short and the result variance is smaller.

ii) nursery:
a. Decision Tree: max_depth = 7
There are 8 features in this dataset and the performance is:
1: 0.708, 2: 0.834, 3: 0.885, 4: 0.907, 5: 0.934, 6: 0.964, 7: 0.976, 8: 0.976
Overall accuracy with max_depth above 7 is the same so I select the smallest number 7 as the max_depth.
b. Random Forest: F_features = 8, tree_number = 200
I tried F_features from 1-8 and found no specific better performance for every number. Thus, I select 8, number of features in dataset, to include all features for consideration and made better decision.
The overall accuracy is already pretty high in decision tree method, and thus the accuracy is almost the same in random forest method. Large tree_number cannot enhance performance and 200 trees are simply enough.

iii) led:
a. Decision Tree: max_depth = 7

There are 7 features in this dataset and the performance is:
1: 0.683, 2: 0.789, 3: 0.828, 4: 0.842, 5: 0.857, 6: 0.852, 7: 0.858
Overall accuracy with max_depth above its maximum 7 is the same and I choose the maximum number 7 for best accuracy.

b. Random Forest: F_features = 7, tree_number = 100

The reason for choosing these two numbers is the same as in nursery. I choose 7 features to include all features for consideration and 100 trees are already enough when there is no significant performance improvement for more trees.

iv) synthetic. social:

a. Decision Tree: max_depth = 8

There are 128 features in this dataset and selecting the max_depth is an interesting choice. The performace is:
1: 0.355, 5: 0.469, 6: 0.49, 7: 0.491, 8: 0.501, 9: 0.484, 10: 0.476, 11: 0.477
The overall accuracy with max_depth above 11 to maxmimum 128 is all the same, and I select 8 for the best accuracy.

b. Random Forest: F_features = 11, tree_number = 300

There are plenty of features in this dataset and select all features as candidate for every node with so many trees can significantly increase the running time. Therefore, I select 11 as the int(square root of 128).
I tried tree_number from 10, 20, 50, 100-500 and find that 300 trees can yield good performance while tree number above 300 cannot enhance performance but increase running time a lot.

3. Model evaluation measures:

For each dataset, each method of decision tree and random forest, and each dataset of training set and test set, there are 7 evaluation features per class and 1 overall accuracy:
7 evaluation features per class are (Using P, N, TP, TN, FP, FN)
Class accuracy= TP+TN/P+N;
Specificity = TN/N;
Precision = TP/TP+FP;
Recall = TP/P;
F-1 Score = 2TP/(2TP+FP+FN);
F-0.5 Score = $(1+0.5^2)$TP/$(1+0.5^2)$TP+0.5^2FN+FP;
F-2 Score = $(1+2^2)$TP/$(1+2^2)$TP+2^2FN+FP;
UNDEF is defined as 0/0.
INF is defined when denominator is 0.
The results are as follows:

1) balance.scale:

Decision Tree:

training set

11	7	9
6	168	12

5 22 160
overall accuracy: 0.8475
Class 1
Accuracy : 0.9325, Specificity : 0.9705, Precision : 0.5, Recall : 0.4074
F-1 Score : 0.4490, 0.5F Score: 0.4783, 2F Score: 0.4231
Class 2
Accuracy : 0.8825, Specificity : 0.8645, Precision : 0.8528, Recall : 0.9032
F-1 Score : 0.8773, 0.5F Score: 0.8624, 2F Score: 0.8927
Class 3
Accuracy : 0.88, Specificity : 0.9014, Precision : 0.8840, Recall : 0.8556
F-1 Score : 0.8696, 0.5F Score: 0.8782, 2F Score: 0.8611

testing set
0 15 7
8 80 14
25 18 58
overall accuracy: 0.6133
Class 1
Accuracy : 0.7556, Specificity : 0.8374, Precision : 0.0, Recall : 0.0
F-1 Score : 0.0, 0.5F Score: 0.0, 2F Score: 0.0
Class 2
Accuracy : 0.7556, Specificity : 0.7317, Precision : 0.7080, Recall : 0.7843
F-1 Score : 0.74419, 0.5F Score: 0.7220, 2F Score: 0.7678
Class 3
Accuracy : 0.7156, Specificity : 0.8306, Precision : 0.7342, Recall : 0.5743
F-1 Score : 0.6444, 0.5F Score: 0.6954, 2F Score: 0.6004

Random Forest:

training set
27 0 0
0 186 0
0 0 187
overall accuracy: 1.0
Class 1
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 2
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 3
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0

testing set
0 15 7
0 92 10
0 16 85

overall accuracy: 0.7867
Class 1
Accuracy : 0.9022, Specificity : 1.0, Precision : UNDEF, Recall : 0.0
F-1 Score : 0.0, 0.5F Score: 0.0, 2F Score: 0.0
Class 2
Accuracy : 0.8178, Specificity : 0.7480, Precision : 0.7480, Recall : 0.9020
F-1 Score : 0.8178, 0.5F Score: 0.7744, 2F Score: 0.8663
Class 3
Accuracy : 0.8533, Specificity : 0.8629, Precision : 0.8333, Recall : 0.8416
F-1 Score : 0.8374, 0.5F Score: 0.8350, 2F Score: 0.8399

2) led:

Decision Tree:
training set
496 142
151 1298
overall accuracy: 0.8596
Class 1
Accuracy : 0.8596, Specificity : 0.8958, Precision : 0.7666, Recall : 0.777
F-1 Score : 0.7720, 0.5F Score: 0.7688, 2F Score: 0.7752
Class 2
Accuracy : 0.8596, Specificity : 0.7774, Precision : 0.9014, Recall : 0.8958
F-1 Score : 0.8986, 0.5F Score: 0.9003, 2F Score: 0.8969

testing set
274 77
84 699
overall accuracy: 0.8580
Class 1
Accuracy : 0.8580, Specificity : 0.8927, Precision : 0.7654, Recall : 0.7806
F-1 Score : 0.7729, 0.5F Score: 0.7684, 2F Score: 0.7775
Class 2
Accuracy : 0.8580, Specificity : 0.7806, Precision : 0.9008, Recall : 0.8927
F-1 Score : 0.8967, 0.5F Score: 0.8992, 2F Score: 0.8943

Random Forest:
training set
489 149
144 1305
overall accuracy: 0.8596
Class 1
Accuracy : 0.8596, Specificity : 0.9006, Precision : 0.7725, Recall : 0.7665
F-1 Score : 0.7695, 0.5F Score: 0.7713, 2F Score: 0.7677
Class 2
Accuracy : 0.8596, Specificity : 0.7665, Precision : 0.8975, Recall : 0.9006
F-1 Score : 0.8991, 0.5F Score: 0.8981, 2F Score: 0.9

testing set
274 77
84 699
overall accuracy: 0.8580
Class 1
Accuracy : 0.8580, Specificity : 0.8927, Precision : 0.7654, Recall : 0.7806
F-1 Score : 0.7729, 0.5F Score: 0.7684, 2F Score: 0.7775
Class 2
Accuracy : 0.8580, Specificity : 0.7806, Precision : 0.9008, Recall : 0.8927
F-1 Score : 0.8967, 0.5F Score: 0.8992, 2F Score: 0.8943

3) nursery:

Decision Tree:
training set
2663 5 2 0 0
3 195 0 0 0
12 0 2496 0 0
0 0 0 2715 0
0 1 0 0 1
overall accuracy: 0.9972
Class 1
Accuracy : 0.9973, Specificity : 0.9972, Precision : 0.9944, Recall : 0.9974
F-1 Score : 0.9959, 0.5F Score: 0.9950, 2F Score: 0.9968
Class 2
Accuracy : 0.9989, Specificity : 0.9992, Precision : 0.97015, Recall : 0.9848
F-1 Score : 0.9774, 0.5F Score: 0.97305, 2F Score: 0.9819
Class 3
Accuracy : 0.9983, Specificity : 0.9996, Precision : 0.9992, Recall : 0.9952
F-1 Score : 0.9972, 0.5F Score: 0.9984, 2F Score: 0.9960
Class 4
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 5
Accuracy : 0.9999, Specificity : 1.0, Precision : 1.0, Recall : 0.5
F-1 Score : 0.6667, 0.5F Score: 0.8333, 2F Score: 0.5556

testing set
1541 42 13 0 0
30 97 0 0 3
34 0 1502 0 0
0 0 0 1605 0
0 0 0 0 0
overall accuracy: 0.9749
Class 1
Accuracy : 0.9755, Specificity : 0.9804, Precision : 0.9601, Recall : 0.96554

F-1 Score : 0.9628, 0.5F Score: 0.9612, 2F Score: 0.9645
Class 2
Accuracy : 0.9846, Specificity : 0.9911, Precision : 0.6978, Recall : 0.74615
F-1 Score : 0.7212, 0.5F Score: 0.7070, 2F Score: 0.7360
Class 3
Accuracy : 0.9903, Specificity : 0.9961, Precision : 0.9914, Recall : 0.9779
F-1 Score : 0.9846, 0.5F Score: 0.9887, 2F Score: 0.9805
Class 4
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 5
Accuracy : 0.9994, Specificity : 0.9994, Precision : 0.0, Recall : UNDEF
F-1 Score : 0.0, 0.5F Score: 0.0, 2F Score: 0.0

Random Forest:

training set

2670	0	0	0	0
0	198	0	0	0
0	0	2508	0	0
0	0	0	2715	0
0	0	0	0	2

overall accuracy: 1.0
Class 1
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 2
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 3
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 4
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 5
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0

testing set

1559	22	15	0	0
35	94	0	0	1
27	0	1509	0	0
0	0	0	1605	0
0	0	0	0	0

overall accuracy: 0.9795
Class 1
Accuracy : 0.9797, Specificity : 0.9810, Precision : 0.9617, Recall : 0.9768

F-1 Score : 0.9692, 0.5F Score: 0.9647, 2F Score: 0.9738
Class 2
Accuracy : 0.98808, Specificity : 0.9954, Precision : 0.8103, Recall : 0.7231
F-1 Score : 0.7642, 0.5F Score: 0.7912, 2F Score: 0.7390
Class 3
Accuracy : 0.9914, Specificity : 0.9955, Precision : 0.9902, Recall : 0.9824
F-1 Score : 0.9863, 0.5F Score: 0.9886, 2F Score: 0.9840
Class 4
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 5
Accuracy : 0.9998, Specificity : 0.9998, Precision : 0.0, Recall : UNDEF
F-1 Score : 0.0, 0.5F Score: 0.0, 2F Score: 0.0

4) synthetic.social:

Decision Tree:

training set
703 11 12 6
16 719 10 10
31 15 717 5
37 20 8 680
overall accuracy: 0.9397
Class 1
Accuracy : 0.96237, Specificity : 0.9630, Precision : 0.8933, Recall : 0.9604
F-1 Score : 0.9256, 0.5F Score: 0.9059, 2F Score: 0.9462
Class 2
Accuracy : 0.9727, Specificity : 0.9795, Precision : 0.9399, Recall : 0.9523
F-1 Score : 0.94605, 0.5F Score: 0.9423, 2F Score: 0.9498
Class 3
Accuracy : 0.973, Specificity : 0.9866, Precision : 0.9598, Recall : 0.9336
F-1 Score : 0.9465, 0.5F Score: 0.9544, 2F Score: 0.9387
Class 4
Accuracy : 0.9713, Specificity : 0.9907, Precision : 0.9700, Recall : 0.91275
F-1 Score : 0.9405, 0.5F Score: 0.9580, 2F Score: 0.9237

testing set
140 28 49 51
35 104 56 50
46 44 126 16
56 49 19 131
overall accuracy: 0.501
Class 1
Accuracy : 0.735, Specificity : 0.8128, Precision : 0.5054, Recall : 0.5224
F-1 Score : 0.5138, 0.5F Score: 0.5087, 2F Score: 0.5189
Class 2

Accuracy : 0.738, Specificity : 0.8397, Precision : 0.4622, Recall : 0.4245
F-1 Score : 0.4426, 0.5F Score: 0.4541, 2F Score: 0.4315
Class 3
Accuracy : 0.77, Specificity : 0.8385, Precision : 0.504, Recall : 0.5431
F-1 Score : 0.5228, 0.5F Score: 0.5114, 2F Score: 0.5348
Class 4
Accuracy : 0.759, Specificity : 0.8440, Precision : 0.5282, Recall : 0.5137
F-1 Score : 0.5209, 0.5F Score: 0.5253, 2F Score: 0.5166

Random Forest:

training set

732	0	0	0
0	755	0	0
0	0	768	0
0	0	0	745

overall accuracy: 1.0
Class 1
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 2
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 3
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0
Class 4
Accuracy : 1.0, Specificity : 1.0, Precision : 1.0, Recall : 1.0
F-1 Score : 1.0, 0.5F Score: 1.0, 2F Score: 1.0

testing set

215	1	24	28
5	188	19	33
20	22	189	1
24	17	3	211

overall accuracy: 0.803
Class 1
Accuracy : 0.898, Specificity : 0.9331, Precision : 0.8144, Recall : 0.8022
F-1 Score : 0.8083, 0.5F Score: 0.8119, 2F Score: 0.8046
Class 2
Accuracy : 0.903, Specificity : 0.9470, Precision : 0.8246, Recall : 0.7673
F-1 Score : 0.7949, 0.5F Score: 0.8124, 2F Score: 0.7781
Class 3
Accuracy : 0.911, Specificity : 0.9401, Precision : 0.8043, Recall : 0.8147
F-1 Score : 0.8094, 0.5F Score: 0.8063, 2F Score: 0.8126
Class 4
Accuracy : 0.894, Specificity : 0.9168, Precision : 0.7729, Recall : 0.8275
F-1 Score : 0.7992, 0.5F Score: 0.7832, 2F Score: 0.8159

4. Comparison and conclusions

Overall accuracy is used to evaluate performance of models

1) balance.scale

DecisionTree: 61.3%; max_depth = 3

RandomForest: ~78% (variance around 1%); F_features = 1, tree_num = 500

Random Forest method largely enhances the performance of decision tree method. The reasons for the improvement may be that I use 500 trees to produce a more comprehensive and robust result. And for small sets of attributes, randomly selecting one may yield better result than using gini index for selection.

2) led

DecisionTree: 85.8%; max_depth = 7

RandomForest: 85.8%(variance around 0.1%); F_features = 7, tree_num = 100

Random Forest method doesn't improve the performance of decision tree method. The reason may be that the TP values(diagonal values in confusion matrix) are dominant and therefore random forest can only add few counts to diagonal counts. The influence may be very slight so that there won't be large improvement

3) nursery

DecisionTree: 97.49%; max_depth = 7

RandomForest: 98% (variance around 0.5%); F_features = 8, tree_num = 200

Random Forest method slightly improves the performance of decision tree method. It's because that the decision tree method already has very high accuracy and diagonal elements in the confusion matrix are dominant. Thus, the accuracy may not be enhanced very much.

4) synthetic.social

DecisionTree: 50.1%; max_depth = 8

RandomForest: 80.3% (variance around 2%); F_features = 11, tree_num = 300

Random Forest method largely improves the performance of decision tree method. The reasons may be that the accuracy of decision tree method is not good enough so that the diagonal values are not dominant. When we use 300 trees to select majority vote, the result can be more accurate and the diagonal values in confusion matrix can be largely increased to improve overall performance.