UCLA STATS 101B

Final Project: Parameter Tuning of Random Forest using Design of Experiments

Lecturer: Alan Vazquez
Teaching Assistants: Zhiqian Zhai and Davis Berlind

1. Background

Data science is an interdisciplinary field that uses algorithms to extract knowledge and insights from data. A relevant problem in this field is *classification*. In this type of problem, a data set consisting of observations from several predictors and a binary response is available. The binary response takes two values (such as 0 or 1) to indicate the classes. The goal is to build an algorithm that correctly classifies the observations in the data set as well as those in new data sets that may be collected in the future. Classification problems include credit card fraud detection, diabetes prediction, and breast cancer prediction.

Random forest is one of the most successful algorithms for tackling classification problems. The algorithm implements specialized statistical methods and techniques, and involves several tuning parameters that affect its performance. **In this project, you will find the most important tuning parameters of random forest that drive the so-called cross-validation accuracy.** To this end, you will plan, conduct and analyze experiments.

The rest of this document gives an introduction to the main elements of random forest and the cross-validation accuracy. It also provides the questions that you must answer in the final project report. To conduct the experiments, we developed an R function called 'cv.rf' that allows you to try different settings of the tuning parameters of random forest and record the cross-validation accuracy. The document called "Tutorial_cvrf.pdf" shows how to use the function.

The document called "General_Guidelines.pdf" shows the evaluation guidelines for the final project.

Random forest

Random forest is an algorithm that classifies observations into one of two possible classes. This algorithm is, however, considered a black-box algorithm which is composed of elaborated and highly-technical techniques. For the final project, you do not need to know all the details behind the algorithm. You only need to know the key idea behind it. Here, we provide a brief overview of the main elements of random forest.

Random forest is built-up from decision (also called classification) trees. Essentially, a decision tree is an algorithm that classifies observations using a set of rules that act on the predictor's values. A classification tree is composed of nodes; see Figure 1. At each node, a splitting criterion based on the predictors is constructed and the observations for which this criterion is *true* are sent to the right child node while instances for which the criteria is *false* are sent to the left child node. The top node of the tree is referred to as the root node while nodes at the bottom of the tree are referred to as terminal nodes.

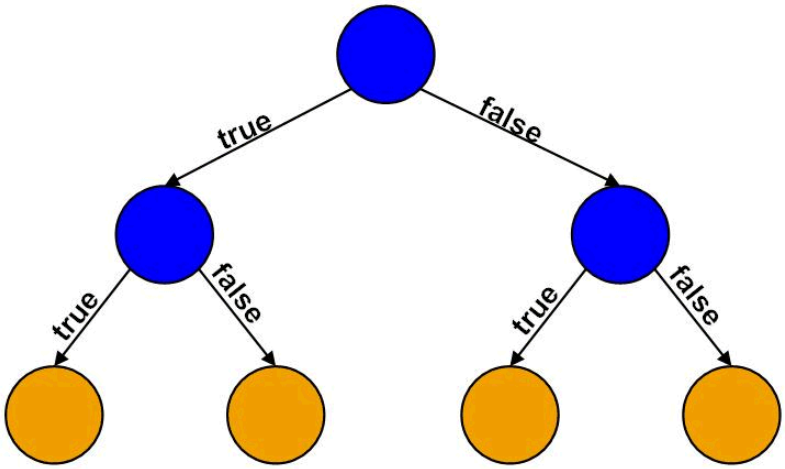

Figure 1. Outline of a decision tree.

Decision trees are built using a data set and a specific method. In the method, it is possible to set the maximum number of nodes in a tree (controlled by the *maxnodes* parameter), the number of observations in each terminal node (controlled by the *nodesize* parameter), and the number of predictors used to build the trees (controlled by the *mtry* parameter).

To classify a new observation (not in the data set), we simply input the predictors' values in the tree and record the classification. It is well-known that a single classification tree does not have the best classification performance for future observations. To overcome this issue, random forest constructs many decision trees from different random samples of the data set at hand, and combines their classifications. More specifically, a random sample with or without replacement (controlled by the *replace* parameter) is selected and a decision tree is built using the sample. The final class assigned by a random forest is the most common or frequent class assigned by the decision trees. This strategy is called majority voting. When constructing a random forest, it is possible to set the total number of trees used (controlled by the *ntree* parameter). Figure 2 shows a schematic of random forest.

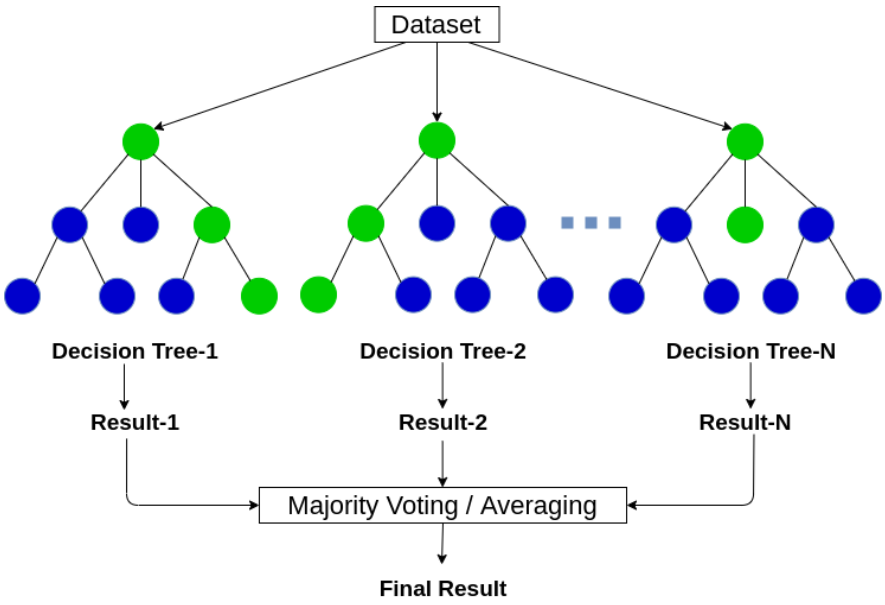

Figure 2. Outline of random forest.

The classification performance of a random forest may be further improved by incorporating additional components to the algorithm. For instance, it is possible to assign different weights to the two classes (controlled by the *classwt* parameter) under study so as to emphasize or understate their correct classification. To this end, it is common to denote the classes as “0” and “1”, and use the later as the reference class. To specify the weights, we just need to set the weight for the reference class, since the weights for both classes must sum to one. It is also possible to change the so-called threshold for classifying observations (controlled by the *cuttoff* parameter). Technically, random forest does not output a class but a probability that the observation belongs to the reference class. A threshold is used to decide if the probability is high enough for the observation to be considered as belonging to the reference class ("1”). Otherwise, it is assigned to the “0” class.

Tuning parameters

In this project, you will investigate the performance of random forest in terms of seven tuning parameters. The parameters are classified into three categories: general, specific to the decision trees, and technical. The parameters are shown below.

General parameters

1. Number of trees (*ntree*).
2. Sampling with or without replacement (*replace*).

Decision trees’ parameters

3. Number of predictors used to build a tree (*mtry*).
4. Number of observations in each terminal node (*nodesize*).

5. Maximum number of nodes in a tree (*maxnodes*).

Technical parameters

6. Prior probability for the reference class (*classwt*).
7. Threshold for binary classification (*cutoff*).

Table 1 shows the feasible settings of these parameters. For the parameter *replace*, a "0" indicates sampling without replacement while a "1" sampling with replacement.

Table 1. Tuning parameters of random forest and their ranges.

Parameter	Low level	High level
ntree	100	1000
replace	0	1
mtry	2	6
nodesize	1	11
maxnodes	10	1000
classwt	0.5	0.9
cutoff	0.2	0.8

Cross-validation accuracy

The best way to measure the classification performance of a random forest is to use the metric called *accuracy*. This metric is the proportion of observations correctly classified by the random forest on a new data set (not used to build the algorithm). A higher accuracy is desired. An issue with computing the accuracy is, however, that it is difficult to find new data sets for a problem.

Cross-validation is a method that overcomes this issue using a clever trick. More specifically, cross-validation splits the data set at hand into two subsets. One subset is used to build a random forest and the other subset to evaluate the algorithm. In this way, we ensure that we have a new data set (not used by random forest) to compute the accuracy. It is recommended to repeat cross-validation several times, each time producing a different split of the data set, a random forest, and accuracy value. The average accuracy among all repetitions of this process is called the cross-validation accuracy, which ranges from 0 to 1.

The goal of this project is to identify the tuning parameters of random forest that affect the cross-validation accuracy.

Auxiliary data sets

To study the performance of random forest, we need auxiliary data sets involving classification problems. In this project, we will use three data sets: **Diabetes Health Indicators**, **Personal Key Indicators of heart Disease**, and **Cardiovascular Disease**. **These data sets will only help us to build a random forest for a classification problem and must not be analyzed.** In this

7 factors

project, the variables under study are the tuning parameters of random forest and the response is the cross-validation accuracy.

We will assign a data set to each team at random. The assignment is shown in the Google spreadsheet you used to register your team.

A brief description of the auxiliary data sets is included in the document called 'Data_Description_Spring_2022.pdf'. The document shows the predictors and response for the classification problem. The data sets are available as R data files: "diabetes.RData", "heart.RData" and "cardiovascular.RData".

2. Task Description

Due to budgetary constraints, the maximum number of tests in the whole experimentation process
35/128 is 35. That is, you can test a maximum of 35 combinations of the settings of the tuning parameters. If you run all the tests on a single computer, consider the experiment as completely randomized.

In your report, you must provide comprehensive answers to the questions below. In your presentation, you must provide a brief summary of your solution to the final project.

Part I: Design of the Experiment

Question 1. Propose a fractional factorial design for the problem. In addition, propose an experimental design constructed using the optimal design approach.

Question 2. Compare the optimal design with the fractional factorial design in practical and statistical terms. For instance, what is the performance of the designs for studying the main effects of the tuning parameters only? Can they estimate all two-parameter interactions? Why or why not? How do they compare in terms of multicollinearity?

Question 3. Recommend one experimental design between the two options in Question 1. Motivate your decision.

Question 4. Using a commercial software, the TAs and I came up with the experimental design shown in Table 2. How does your recommended design in the previous question compare with this one?

Table 2. Alternative experimental design.

Run	ntree	replace	mtry	nodesize	maxnodes	classwt	cutoff
1	100	1	2	11	10	0.5	0.8
2	550	0	2	1	10	0.5	0.2
3	1000	1	4	1	10	0.5	0.2
4	1000	1	6	1	1000	0.5	0.8
5	1000	0	6	1	1000	0.9	0.2
6	100	0	2	1	1000	0.5	0.8
7	1000	0	2	6	10	0.9	0.8
8	100	0	2	11	10	0.9	0.2
9	100	1	6	1	10	0.7	0.8
10	100	0	6	1	10	0.9	0.5
11	100	0	4	11	1000	0.9	0.8
12	1000	1	2	11	1000	0.5	0.5
13	100	1	6	6	1000	0.5	0.2
14	550	0	4	6	505	0.7	0.5
15	100	0	6	11	505	0.5	0.2
16	1000	0	6	11	10	0.5	0.8
17	1000	0	2	11	1000	0.7	0.2
18	1000	1	6	11	10	0.9	0.2
19	100	1	2	1	1000	0.9	0.2
20	1000	1	2	1	505	0.9	0.8
21	550	1	4	6	505	0.7	0.5
22	550	1	6	11	1000	0.9	0.8

Part II: Data Analysis

Question 5. Collect data using your recommended design in Question 3.

Question 6. Conduct a detailed data analysis. What are the influential tuning parameters? What is the final model that links the tuning parameters to the cross-validation accuracy? Does the final model provide a good fit to the data?