IST438-W8-Applications

4/24/23

Ensemble Learning: Bagging trees and random forests

In this application, we will practice on tree-based ensemble learning models such as bagging trees and random forests using `{ranger}` package. It provides useful functions for faster implementation of random forests.

```
# install.packages("ranger")
library(ranger)
```

Dataset

The `PimaIndiansDiabetes` data set as it relates to predicting whether someone has diabetes. This data is provided by the mlbench package.

```
#install.packages("mlbench")
library(mlbench)
data("PimaIndiansDiabetes")
str(PimaIndiansDiabetes)
```

```
'data.frame':   768 obs. of  9 variables:
 $ pregnant: num  6 1 8 1 0 5 3 10 2 8 ...
 $ glucose : num  148 85 183 89 137 116 78 115 197 125 ...
 $ pressure: num  72 66 64 66 40 74 50 0 70 96 ...
 $ triceps : num  35 29 0 23 35 0 32 0 45 0 ...
 $ insulin : num  0 0 0 94 168 0 88 0 543 0 ...
 $ mass    : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
 $ pedigree: num  0.627 0.351 0.672 0.167 2.288 ...
 $ age     : num  50 31 32 21 33 30 26 29 53 54 ...
```

```
 $ diabetes: Factor w/ 2 levels "neg","pos": 2 1 2 1 2 1 2 1 2 2 ...
```

The relevant variables are:

`pregnant` - Number of times pregnant

`glucose` - Plasma glucose concentration (glucose tolerance test)

`pressure` - Diastolic blood pressure (mm Hg)

`triceps` - Triceps skin fold thickness (mm)

`insulin` - 2-Hour serum insulin (mu U/ml)

`mass` - Body mass index (weight in kg/(height in m)^2)

`pedigree` - Diabetes pedigree function

`age` - Age (years)

`diabetes` - Class variable (test for diabetes)

Splitting

```
library(rsample)
diabetes_split <- initial_split(data = PimaIndiansDiabetes, # dataset to split
                                prop = 0.80)    # proportion of train set

diabetes_train <- diabetes_split |> training()
diabetes_test  <- diabetes_split |> testing()
```

Training bagging trees

`{ranger}` package has a main function `ranger()` to train bagging trees and random forests. When you set the `mtry` argument is equal to the number of features, the function returns a trained bagging trees.

```
trained_bt <- ranger(diabetes ~ .,
                     data = diabetes_train,
                     mtry = 8)
```

Let's see the ouput of the model object:

```
trained_bt
```

```
Ranger result

Call:
 ranger(diabetes ~ ., data = diabetes_train, mtry = 8)

Type:                             Classification
Number of trees:                  500
Sample size:                      614
Number of independent variables:  8
Mtry:                             8
Target node size:                 1
Variable importance mode:         none
Splitrule:                        gini
OOB prediction error:             25.24 %
```

It is seen that the output returns (1) model formula, (2) the values of hyperparameters, and (3) prediction error.

Training random forests

```
trained_rf <- ranger(diabetes ~ .,
                     data = diabetes_train)
```

Let's see the ouput of the model object:

```
trained_rf
```

```
Ranger result

Call:
 ranger(diabetes ~ ., data = diabetes_train)

Type:                             Classification
Number of trees:                  500
Sample size:                      614
Number of independent variables:  8
```

```
Mtry:                           2
Target node size:               1
Variable importance mode:       none
Splitrule:                      gini
OOB prediction error:           23.94 %
```

The output is in same type with the bagging trees' one. There are only two differences in the output: (1) the value of `mtry`, and (2) the value of OOB prediction error.

The value of `mtry` is calculated according to the formula: $\sqrt{p} = \sqrt{8} \sim 2$ for classification task.