4_3.R

ramom

2023-06-27

```
#setting up working directory
getwd()
```

```
## [1] "C:/Users/ramom/Desktop/MDS/Academic/1st Semester/MDS- 3- R/Assignments/Assignment4.3"
```

```
setwd("C:/Users/ramom/Desktop/MDS/Academic/1st Semester/MDS- 3- R/Assignments/Assignment4.3")
getwd()
```

```
## [1] "C:/Users/ramom/Desktop/MDS/Academic/1st Semester/MDS- 3- R/Assignments/Assignment4.3"
```

```
#load data from csv file
data <- read.csv("titanic.csv")
head(data)
```

```
##   Survived Pclass                                                Name    Sex Age Siblings.Spouses.Abo
## 1        0      3                             Mr. Owen Harris Braund   male  22
## 2        1      1 Mrs. John Bradley (Florence Briggs Thayer) Cumings female  38
## 3        1      3                              Miss. Laina Heikkinen female  26
## 4        1      1        Mrs. Jacques Heath (Lily May Peel) Futrelle female  35
## 5        0      3                           Mr. William Henry Allen   male  35
## 6        0      3                                   Mr. James Moran   male  27
```

```
#remove name column by index
data <- data[, -3]
str(data)
```

```
## 'data.frame':    887 obs. of  7 variables:
##  $ Survived               : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass                 : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Sex                    : chr  "male" "female" "female" "female" ...
##  $ Age                    : num  22 38 26 35 35 27 54 2 27 14 ...
##  $ Siblings.Spouses.Aboard: int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parents.Children.Aboard: int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Fare                   : num  7.25 71.28 7.92 53.1 8.05 ...
```

```
#2. Fit binary logistic regression model with "Survived" variable as dependent
#variable and rest of variables as independent variables using "data",
#get summary of the model, check VIF and interpret the results carefully

#Fit binary logistic regression model
Binary_Logistic_reg <- glm(Survived ~ ., data = data, family = binomial)
summary(Binary_Logistic_reg)  #summary of the model
```

```
##
## Call:
## glm(formula = Survived ~ ., family = binomial, data = data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7789  -0.5976  -0.3987   0.6156   2.4409
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               5.297252   0.557409   9.503  < 2e-16 ***
## Pclass                   -1.177659   0.146079  -8.062 7.52e-16 ***
## Sexmale                  -2.757282   0.200416 -13.758  < 2e-16 ***
## Age                      -0.043474   0.007723  -5.629 1.81e-08 ***
## Siblings.Spouses.Aboard -0.401831   0.110712  -3.630 0.000284 ***
## Parents.Children.Aboard -0.106505   0.118588  -0.898 0.369127
## Fare                      0.002786   0.002389   1.166 0.243680
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.77  on 886  degrees of freedom
## Residual deviance:  780.93  on 880  degrees of freedom
## AIC: 794.93
##
## Number of Fisher Scoring iterations: 5
```

```
# # Generate predictions on the training data
# predicted <- predict(Binary_Logistic_reg, newdata = data, type = "response")
# predicted <- ifelse(predicted >= 0.5, 1, 0)
#
# # Compute confusion matrix on the training data
# confusion_matrix_train <- table(data$Survived, predicted)
# print(confusion_matrix_train)
#
# # Compute accuracy on the training data
#accuracy_train <- sum(diag(confusion_matrix_train)) / sum(confusion_matrix_train)
#misclassification_error_train <- 1 - accuracy_train
#print(paste("Accuracy (Training):", accuracy_train))
#print(paste("Misclassification Error (Training):", misclassification_error_train))

library(car)
vif(Binary_Logistic_reg)
```

```
##                  Pclass                     Sex                     Age Siblings.Spouses.Aboard Parer
##                1.886336                1.198648                1.452160                1.288629
##                    Fare
##                1.482601
```

```
#3. Randomly split the data into 70% and 30% with replacement of
#samples as "train" and "test" data
```

```r
#converting pClass and  sex to the factor variable
data$Pclass <- as.factor(data$Pclass)
data$Sex <- as.factor(data$Sex)

# Perform one-hot encoding for "Pclass"
encoded_pclass <- model.matrix(~ Pclass - 1, data = data)

# Perform one-hot encoding for "Sex"
encoded_sex <- model.matrix(~ Sex - 1, data = data)

# Combine the encoded columns with the original data
data_encoded <- cbind(data[, -which(names(data) %in% c("Pclass", "Sex"))], encoded_pclass, encoded_sex)

#checking
head(data_encoded)
```

```
##   Survived Age Siblings.Spouses.Aboard Parents.Children.Aboard    Fare Pclass1 Pclass2 Pclass3 Sexfe
## 1        0  22                       1                       0  7.2500       0       0       1
## 2        1  38                       1                       0 71.2833       1       0       0
## 3        1  26                       0                       0  7.9250       0       0       1
## 4        1  35                       1                       0 53.1000       1       0       0
## 5        0  35                       0                       0  8.0500       0       0       1
## 6        0  27                       0                       0  8.4583       0       0       1
```

```r
str(data_encoded)
```

```
## 'data.frame':    887 obs. of  10 variables:
##  $ Survived               : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Age                    : num  22 38 26 35 35 27 54 2 27 14 ...
##  $ Siblings.Spouses.Aboard: int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parents.Children.Aboard: int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Fare                   : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Pclass1                : num  0 1 0 1 0 0 1 0 0 0 ...
##  $ Pclass2                : num  0 0 0 0 0 0 0 0 0 1 ...
##  $ Pclass3                : num  1 0 1 0 1 1 0 1 1 0 ...
##  $ Sexfemale              : num  0 1 1 1 0 0 0 0 1 1 ...
##  $ Sexmale                : num  1 0 0 0 1 1 1 1 0 0 ...
```

```r
set.seed(26)
#split the data set
RSplit <- sample(2, nrow(data), replace = T, prob = c(0.7,0.3))
data_train <- data[RSplit == 1, ]
data_test <- data[RSplit == 2, ]

#4. Fit binary logistic regression classifier, knn classifier, ann classifier,
#naive bayes classifier, svm classifier, decision tree classifier,
#decision tree bagging classifier, random forest classifier, tuned random forest
#classifier and random forest boosting classifier models using the "train" data

#Fit binary logistic regression classifier using train data
blrc_Model <- glm(Survived ~ ., data = data_train, family = binomial)
summary(blrc_Model)
```

```
##
## Call:
## glm(formula = Survived ~ ., family = binomial, data = data_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7634  -0.6252  -0.4093   0.6446   2.4068
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)                4.041548   0.543502   7.436 1.04e-13 ***
## Pclass2                   -1.161450   0.354533  -3.276  0.00105 **
## Pclass3                   -2.306297   0.359672  -6.412 1.43e-10 ***
## Sexmale                   -2.589435   0.237510 -10.902  < 2e-16 ***
## Age                       -0.044715   0.009069  -4.931 8.19e-07 ***
## Siblings.Spouses.Aboard -0.378897   0.124938  -3.033  0.00242 **
## Parents.Children.Aboard -0.142208   0.145679  -0.976  0.32898
## Fare                       0.003347   0.003407   0.983  0.32582
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 826.99  on 614  degrees of freedom
## Residual deviance: 560.51  on 607  degrees of freedom
## AIC: 576.51
##
## Number of Fisher Scoring iterations: 5
```

```
# Generate predictions on the training data with classification model
predicted_blrc_training <- predict(blrc_Model, newdata = data_train, type = "response")
predicted_blrc_model <- ifelse(predicted_blrc_training>= 0.5, 1, 0)

#create confusion matrix, calculate accuracy and error
(cm_blrc <- table(predicted_blrc_model, data_train$Survived))
```

```
##
## predicted_blrc_model   0   1
##                    0 320  73
##                    1  50 172
```

```
(accuracy_blrc_training <- sum(diag(cm_blrc))/sum(cm_blrc))
```

```
## [1] 0.8
```

```
(error_blrc_training <- 1 - accuracy_blrc_training)
```

```
## [1] 0.2
```

```
# Generate predictions on the test data
predicted_blrc_testing <- predict(blrc_Model, newdata = data_test, type = "response")
```

```r
predicted_blrc_test_model <- ifelse(predicted_blrc_testing>= 0.5, 1, 0)

#create confusion matrix, calculate accuracy and error
(cm_blrc_test <- table(predicted_blrc_test_model, data_test$Survived))
```

```
##
## predicted_blrc_test_model   0   1
##                         0 152  28
##                         1  23  69
```

```r
(accuracy_blrc_testing <- sum(diag(cm_blrc_test))/sum(cm_blrc_test))
```

```
## [1] 0.8125
```

```r
(error_blrc_testing <- 1 - accuracy_blrc_testing)
```

```
## [1] 0.1875
```

```r
#FITTING THE KNN CLASSIFIER
library(class)
data <- data_encoded #knn use all the encoded data where we
#split factors to individual variables

set.seed(26)
#split the data set with encoded data
RSplit <- sample(2, nrow(data), replace = T, prob = c(0.7,0.3))
data_train <- data[RSplit == 1, ]
data_test <- data[RSplit == 2, ]

# Scale the features
train_data_scaled <- as.data.frame(scale(data_train[, -1]))
test_data_scaled <- as.data.frame(scale(data_test[, -1]))

#fit the knn classifier
knn_model <- knn(train = train_data_scaled,
                 test = test_data_scaled, data_train$Survived, k = 3)
#get the summary
summary(knn_model)
```

```
##   0   1
## 178  94
```

```r
# Compute confusion matrix and accuracy for the classifier model
(knn_model_cm <- table(knn_model, data_test$Survived))
```

```
##
## knn_model   0   1
##         0 153  25
##         1  22  72
```

```r
(accuracy_knn <- sum(diag(knn_model_cm)) / sum(knn_model_cm))
```

```
## [1] 0.8272059
```

```r
(misclassification_error_knn <- 1 - accuracy_knn)
```

```
## [1] 0.1727941
```

```r
# Generate predictions on the training data
knn_predicted_train_model <- knn(scale(data_train[, -1]),
                                 scale(data_train[, -1]),
                                 data_train$Survived, k = 3)

# Compute confusion matrix and accuracy for training data
(knn_predicted_train_cm <- table(knn_predicted_train_model,data_train$Survived))
```

```
##
## knn_predicted_train_model   0   1
##                         0 348  57
##                         1  22 188
```

```r
(accuracy_knn_training <- sum(diag(knn_predicted_train_cm) /
                              sum(knn_predicted_train_cm)))
```

```
## [1] 0.8715447
```

```r
(misclassification_error_knn_training <- 1 - accuracy_knn_training)
```

```
## [1] 0.1284553
```

```r
# Generate predictions on the test data
knn_predicted_test_model <- knn(scale(data_train[, -1]), scale(data_test[, -1]),
                                data_train$Survived, k = 3)

# Compute confusion matrix and accuracy for training data
(knn_predicted_test_cm <- table(knn_predicted_test_model, data_test$Survived))
```

```
##
## knn_predicted_test_model   0   1
##                        0 153  25
##                        1  22  72
```

```r
(accuracy_knn_testing <- sum(diag(knn_predicted_test_cm) /
                             sum(knn_predicted_test_cm)))
```

```
## [1] 0.8272059
```

```r
(misclassification_error_knn_testing <- 1 - accuracy_knn_testing)
```

```
## [1] 0.1727941
```

```r
#FITTING THE ANN MODEL CLASSIFIER
library(neuralnet)

# Fit the ANN classifier model
ann_model <- neuralnet(Survived ~ ., data = data_train, hidden = 5,
                       linear.output = FALSE)
summary(ann_model)
```

```
##                     Length Class      Mode
## call                   5   -none-     call
## response             615   -none-     numeric
## covariate           5535   -none-     numeric
## model.list             2   -none-     list
## err.fct                1   -none-     function
## act.fct                1   -none-     function
## linear.output          1   -none-     logical
## data                  10   data.frame list
## exclude                0   -none-     NULL
## net.result             1   -none-     list
## weights                1   -none-     list
## generalized.weights    1   -none-     list
## startweights           1   -none-     list
## result.matrix         59   -none-     numeric
```

```r
# Generate predictions on the training data
ann_predicted_training <- compute(ann_model, data_train[, -1])$net.result
ann_predicted_training <- ifelse(ann_predicted_training >= 0.5, 1, 0)

# Compute confusion matrix and accuracy for training data
(ann_confusion_matrix_training <- table(ann_predicted_training,
                                        data_train$Survived))
```

```
##
## ann_predicted_training   0   1
##                      0 356  84
##                      1  14 161
```

```r
(ann_accuracy_training <- sum(diag(ann_confusion_matrix_training )) /
    sum(ann_confusion_matrix_training))
```

```
## [1] 0.8406504
```

```r
(ann_misclassification_error_training <- 1 - ann_accuracy_training)
```

```
## [1] 0.1593496
```

```r
# Generate predictions on the testing data
ann_predicted_testing <- compute(ann_model, data_test[, -1])$net.result
ann_predicted_testing <- ifelse(ann_predicted_testing >= 0.5, 1, 0)

# Compute confusion matrix and accuracy for testing data
(ann_confusion_matrix_testing <- table(ann_predicted_testing,
                                       data_test$Survived))
```

```
##
## ann_predicted_testing   0   1
##                     0 163  37
##                     1  12  60
```

```r
(ann_accuracy_testing <- sum(diag(ann_confusion_matrix_testing )) /
    sum(ann_confusion_matrix_testing))
```

```
## [1] 0.8198529
```

```r
(ann_misclassification_error_testing <- 1 - ann_accuracy_testing)
```

```
## [1] 0.1801471
```

```r
#FITTING THE NAIVE BAYES CLASSIFIER
library(e1071)
# Fit the Naive Bayes classifier
naive_bayes_model <- naiveBayes(Survived ~ ., data = data_train)
summary(naive_bayes_model)
```

```
##           Length Class  Mode
## apriori   2      table  numeric
## tables    9      -none- list
## levels    2      -none- character
## isnumeric 9      -none- logical
## call      4      -none- call
```

```r
# Make predictions on the training data
nb_predicted_train <- predict(naive_bayes_model, data_train[, -1])

# Compute confusion matrix and accuracy for training data
(nb_confusion_matrix_training <- table(nb_predicted_train,
                                       data_train$Survived))
```

```
##
## nb_predicted_train   0   1
##                  0 306  74
##                  1  64 171
```

```r
(nb_accuracy_train <- sum(diag(nb_confusion_matrix_training)) /
    sum(nb_confusion_matrix_training))
```

```
## [1] 0.7756098
```

```r
(nb_misclassification_error_train <- 1 - nb_accuracy_train)
```

```
## [1] 0.2243902
```

```r
# Make predictions on the test data
nb_predicted_test <- predict(naive_bayes_model, data_test[, -1])

# Compute confusion matrix and accuracy for testing data
(nb_confusion_matrix_testing <- table(nb_predicted_test, data_test$Survived))
```

```
##
## nb_predicted_test   0   1
##                 0 152  24
##                 1  23  73
```

```r
(nb_accuracy_test <- sum(diag(nb_confusion_matrix_testing)) /
    sum(nb_confusion_matrix_testing))
```

```
## [1] 0.8272059
```

```r
(nb_misclassification_error_test <- 1 - nb_accuracy_test)
```

```
## [1] 0.1727941
```

```r
#FITTING THE SUPPORT VECTOR MACHINE
# Support Vector Machine (SVM) classifier
svm_model <- svm(formula = Survived ~ ., data = data_train,
                 type = 'C-classification',
                 kernel = 'radial')  #use euclidean distance to calculate
#similarities between points
#summary of the model
summary(svm_model)
```

```
##
## Call:
## svm(formula = Survived ~ ., data = data_train, type = "C-classification", kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  296
##
##  ( 149 147 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

```r
# Make predictions on the training data
svm_predicted_train <- predict(svm_model, data_train[, -1])

# Compute confusion matrix and accuracy for training data
(svm_confusion_matrix_training <- table(svm_predicted_train,
                                        data_train$Survived))
```

```
##
## svm_predicted_train   0   1
##                   0 341  83
##                   1  29 162
```

```r
(svm_accuracy_train <- sum(diag(svm_confusion_matrix_training)) /
    sum(svm_confusion_matrix_training))
```

```
## [1] 0.8178862
```

```r
(svm_misclassification_error_train <- 1 - svm_accuracy_train)
```

```
## [1] 0.1821138
```

```r
# Make predictions on the test data
svm_predicted_test <- predict(svm_model, data_test[, -1])

# Compute confusion matrix and accuracy for testing data
(svm_confusion_matrix_testing <- table(svm_predicted_test, data_test$Survived))
```

```
##
## svm_predicted_test   0   1
##                   0 160  27
##                   1  15  70
```

```r
(svm_accuracy_test <- sum(diag(svm_confusion_matrix_testing)) /
    sum(svm_confusion_matrix_testing))
```

```
## [1] 0.8455882
```

```r
(svm_misclassification_error_test <- 1 - svm_accuracy_test)
```

```
## [1] 0.1544118
```

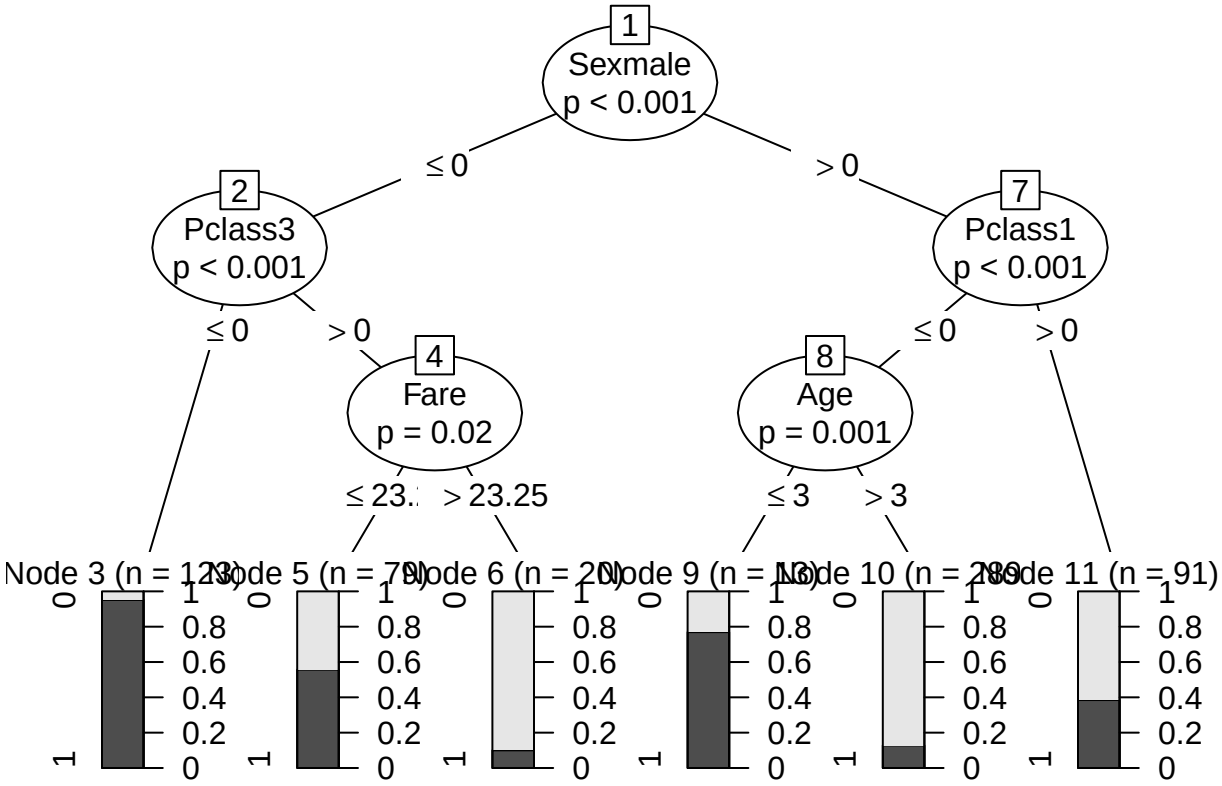
```r
#FITTING DECISION TREE CLASSIFIER
# loading the required library
library(party)
#library(rpart)

data_train$Survived <- as.factor(data_train$Survived)
data_test$Survived <- as.factor(data_test$Survived)
```

```r
# Fit the Decision Tree classifier
# dt_model <- rpart(Survived ~ ., data = data_train, method = "class")
# dt_model
dt_tree_model <- ctree(Survived ~ ., data = data_train)
dt_tree_model
```

```
##
##   Conditional inference tree with 6 terminal nodes
##
## Response:  Survived
## Inputs:  Age, Siblings.Spouses.Aboard, Parents.Children.Aboard, Fare, Pclass1, Pclass2, Pclass3, Sex
## Number of observations:  615
##
## 1) Sexmale <= 0; criterion = 1, statistic = 163.241
##   2) Pclass3 <= 0; criterion = 1, statistic = 66.25
##     3)*  weights = 123
##   2) Pclass3 > 0
##     4) Fare <= 23.25; criterion = 0.98, statistic = 9.365
##       5)*  weights = 79
##     4) Fare > 23.25
##       6)*  weights = 20
## 1) Sexmale > 0
##   7) Pclass1 <= 0; criterion = 1, statistic = 22.15
##     8) Age <= 3; criterion = 0.999, statistic = 14.703
##       9)*  weights = 13
##     8) Age > 3
##       10)*  weights = 289
##   7) Pclass1 > 0
##     11)*  weights = 91
```

```r
# Visualize the decision tree
plot(dt_tree_model)
```




```
# Make predictions on the training data
dt_predicted_train <- predict(dt_tree_model, newdata = data_train)

# Compute confusion matrix and accuracy for training data
(dt_confusion_matrix_train <- table(dt_predicted_train, data_train$Survived))
```

```
##
## dt_predicted_train   0   1
##                  0 326  74
##                  1  44 171
```

```
(dt_accuracy_train <- sum(diag(dt_confusion_matrix_train)) /
    sum(dt_confusion_matrix_train))
```

```
## [1] 0.8081301
```

```
(dt_misclassification_error_train <- 1 - dt_accuracy_train)
```

```
## [1] 0.1918699
```

```
# Make predictions on the testing data
dt_predicted_test <- predict(dt_tree_model, newdata = data_test)

# Compute confusion matrix and accuracy for testing data
(dt_confusion_matrix_test <- table(dt_predicted_test, data_test$Survived))
```

```
##
## dt_predicted_test   0   1
##                 0 157  26
##                 1  18  71
```

```
(dt_accuracy_test <- sum(diag(dt_confusion_matrix_test)) /
    sum(dt_confusion_matrix_test))
```

```
## [1] 0.8382353
```

```
(dt_misclassification_error_test <- 1 - dt_accuracy_test)
```

```
## [1] 0.1617647
```

```
#FITTING THE DECiSION TREE BAGGING CLASSIFIER
library(ipred)
dt_bagging_model <- bagging(Survived ~ ., data = data_train, coob = T)
print(dt_bagging_model)
```

```
##
## Bagging classification trees with 25 bootstrap replications
##
## Call: bagging.data.frame(formula = Survived ~ ., data = data_train,
##     coob = T)
##
## Out-of-bag estimate of misclassification error:  0.226
```

```
# Make predictions on the training data
dt_bagging_predicted_train <- predict(dt_bagging_model, newdata = data_train)

# Compute confusion matrix and accuracy for training data
(dt_bag_confusion_matrix_train <- table(dt_bagging_predicted_train,
                                        data_train$Survived))
```

```
##
## dt_bagging_predicted_train   0   1
##                          0 365  11
##                          1   5 234
```

```
(dt_bag_accuracy_train <- sum(diag(dt_bag_confusion_matrix_train)) /
    sum(dt_bag_confusion_matrix_train))
```

```
## [1] 0.9739837
```

```
(dt_bag_misclassification_error_train <- 1 - dt_bag_accuracy_train)
```

```
## [1] 0.02601626
```

```r
# Make predictions on the testing data
dt_bagging_predicted_test <- predict(dt_bagging_model, newdata = data_test)

# Compute confusion matrix and accuracy for testing data
(dt_bag_confusion_matrix_test <- table(dt_bagging_predicted_test,
                                       data_test$Survived))
```

```
##
## dt_bagging_predicted_test   0   1
##                         0 151  21
##                         1  24  76
```

```r
(dt_bag_accuracy_test <- sum(diag(dt_bag_confusion_matrix_test)) /
    sum(dt_bag_confusion_matrix_test))
```

```
## [1] 0.8345588
```

```r
(dt_bag_misclassification_error_test <- 1 - dt_bag_accuracy_test)
```

```
## [1] 0.1654412
```

```r
#FITTING RANDOM FOREST CLASSIFIER
library(randomForest)
rf_model <- randomForest(Survived ~ ., data = data_train, ntree = 100)
rf_model
```

```
##
## Call:
##  randomForest(formula = Survived ~ ., data = data_train, ntree = 100)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 18.86%
## Confusion matrix:
##     0   1 class.error
## 0 335  35  0.09459459
## 1  81 164  0.33061224
```

```r
# Make predictions on the training data
rf_predicted_train <- predict(rf_model, newdata = data_train)

# Compute confusion matrix and accuracy for training data
(rf_confusion_matrix_train <- table(rf_predicted_train, data_train$Survived))
```

```
##
## rf_predicted_train   0   1
##                  0 355  55
##                  1  15 190
```

```r
(rf_accuracy_train <- sum(diag(rf_confusion_matrix_train)) /
    sum(rf_confusion_matrix_train))
```

```
## [1] 0.8861789
```

```r
(rf_misclassification_error_train <- 1 - rf_accuracy_train)
```

```
## [1] 0.1138211
```

```r
# Make predictions on the testing data
rf_predicted_test <- predict(rf_model, newdata = data_test)

# Compute confusion matrix and accuracy for testing data
(rf_confusion_matrix_test <- table(rf_predicted_test, data_test$Survived))
```

```
##
## rf_predicted_test   0   1
##                 0 160  24
##                 1  15  73
```

```r
(rf_accuracy_test <- sum(diag(rf_confusion_matrix_test)) /
    sum(rf_confusion_matrix_test))
```

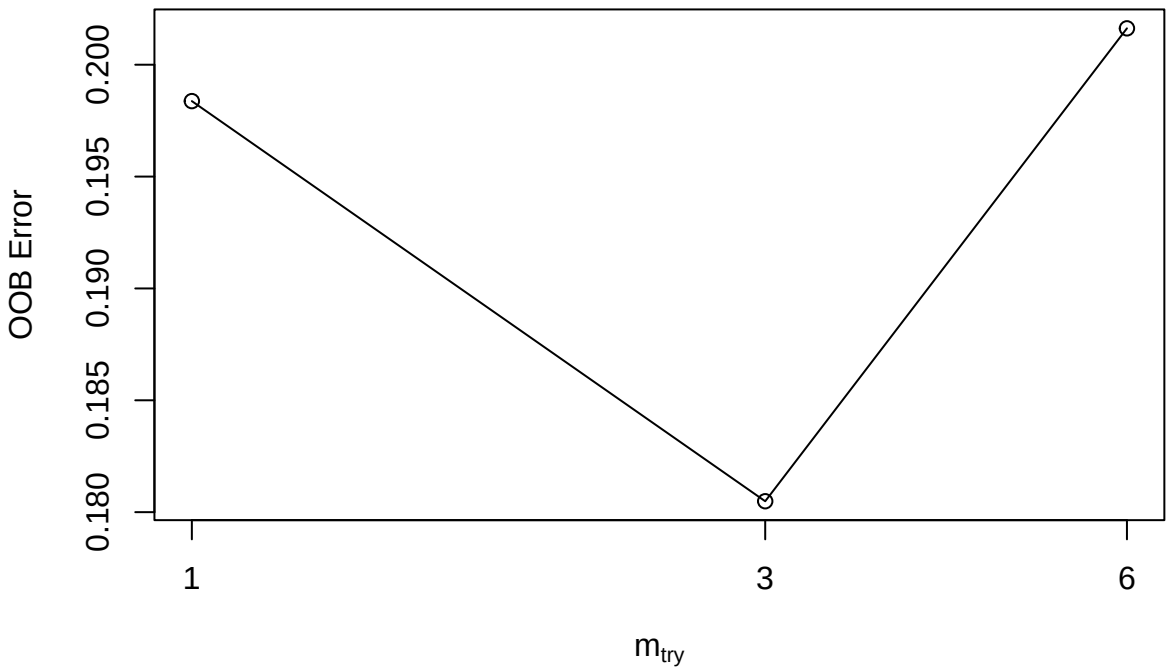
```
## [1] 0.8566176
```

```r
(rf_misclassification_error_test <- 1 - rf_accuracy_test)
```

```
## [1] 0.1433824
```

```r
#FITTING TUNED RANDOM FOREST CLASSIFIER
# Tune the number of variables using tuneRF first before fitting itinto model
mtry_tuned <- tuneRF(
  x = data_train[, -1],  # Exclude the target variable
  y = data_train$Survived,
  ntree = 300,
  stepFactor = 0.5,
  improve = 0.05,
  plot = TRUE
)
```

```
## mtry = 3  OOB error = 18.05%
## Searching left ...
## mtry = 6     OOB error = 20.16%
## -0.1171171 0.05
## Searching right ...
## mtry = 1     OOB error = 19.84%
## -0.0990991 0.05
```




```
# Fit the Random Forest classifier with tuned parameters
trf_model <- randomForest(
  Survived ~ .,
  data = data_train,
  mtry = mtry_tuned,
  ntree = 300
)
```

```
## Warning in mtry < 1 || mtry > p: 'length(x) = 6 > 1' in coercion to 'logical(1)'
```

```
## Warning in mtry < 1 || mtry > p: 'length(x) = 6 > 1' in coercion to 'logical(1)'
```

```
# Make predictions on the training data
trf_predicted_train <- predict(trf_model, newdata = data_train)

# Compute confusion matrix and accuracy for training data
(trf_confusion_matrix_train <- table(trf_predicted_train, data_train$Survived))
```

```
##
## trf_predicted_train   0   1
##                   0 356  88
##                   1  14 157
```

```r
(trf_accuracy_train <- sum(diag(trf_confusion_matrix_train)) /
    sum(trf_confusion_matrix_train))
```

```
## [1] 0.8341463
```

```r
(trf_misclassification_error_train <- 1 - trf_accuracy_train)
```

```
## [1] 0.1658537
```

```r
# Make predictions on the test data
trf_predicted_test <- predict(trf_model, newdata = data_test)

# Compute confusion matrix and accuracy for testing data
(trf_confusion_matrix_test <- table(trf_predicted_test, data_test$Survived))
```

```
##
## trf_predicted_test   0   1
##                  0 170  44
##                  1   5  53
```

```r
(trf_accuracy_test <- sum(diag(trf_confusion_matrix_test)) /
    sum(trf_confusion_matrix_test))
```

```
## [1] 0.8198529
```

```r
(trf_misclassification_error_test <- 1 - trf_accuracy_test)
```

```
## [1] 0.1801471
```

```r
#FITTING RANDOM FOREST BOOSTING CLASSIFIER
library(xgboost)
library(caret)

# Define the training control parameters
control <- trainControl(
  method = "cv",  # Cross-validation resampling method
  number = 10,  # Number of cross-validation folds
  verboseIter = TRUE,  # Print verbose output
  allowParallel = TRUE  # Enable parallel processing if available
)

# Define the parameter grid for tuning
param_grid <- expand.grid(
  nrounds = c(50, 100, 150),  # Number of boosting iterations
  max_depth = c(3, 4, 5),  # Maximum tree depth
  eta = c(0.1, 0.2, 0.3),  # Learning rate
  gamma = 0,  # Minimum loss reduction required to make a further partition on a leaf node
  colsample_bytree = 1,  # Subsample ratio of columns when constructing each tree
  min_child_weight = 1,  # Minimum sum of instance weight (hessian) needed in a child
  subsample = 1  # Subsample ratio of the training instances
)
```

```
)

#Fit the boosting of random forest model using cross-validation&parameter tuning
boosting_rf_model <- train(
  Survived ~ .,
  data = data_train,
  method = "xgbTree",
  trControl = control,
  tuneGrid = param_grid
)
```

```
## + Fold01: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:41] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:41] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:42] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:42] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:42] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:42] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:44] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:44] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:44] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:44] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold01: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:44] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:44] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold01: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:45] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:45] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:45] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
```

```
## [13:03:45] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:46] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:46] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:46] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:46] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:46] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:46] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold02: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:48] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:48] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold02: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:48] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:48] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:48] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:48] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:49] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:49] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:49] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:49] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:49] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:49] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:50] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:50] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
```

```
## + Fold03: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:50] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:50] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold03: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold03: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:53] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:53] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:53] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:53] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:54] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:54] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:54] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:54] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold04: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:54] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:54] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold04: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:55] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:55] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:55] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
```

```
## [13:03:55] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:55] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:55] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:56] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:56] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:56] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:56] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:57] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:57] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:57] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:57] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:57] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:57] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold05: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold05: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:03:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:03:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:00] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:00] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
```

```
## + Fold06: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:00] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:00] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:00] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:00] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold06: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold06: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:02] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:02] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:02] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:02] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold07: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold07: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:05] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
```

```
## [13:04:05] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:05] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:05] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:05] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:05] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:06] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:06] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:06] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:06] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:07] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:07] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:07] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:07] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold08: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:07] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:07] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold08: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:08] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:08] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:08] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:08] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:09] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:09] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:09] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:09] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:09] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:09] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:10] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:10] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
```

```
## + Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:10] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:10] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:10] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:10] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold09: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:11] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:11] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold09: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:11] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:11] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.1, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:11] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:11] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.1, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:12] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:12] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.1, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:12] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:12] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.2, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:12] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:12] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.2, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:13] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:13] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.2, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:13] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:13] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.3, max_depth=3, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:14] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:14] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.3, max_depth=4, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## + Fold10: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## [13:04:14] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:04:14] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## - Fold10: eta=0.3, max_depth=5, gamma=0, colsample_bytree=1, min_child_weight=1, subsample=1, nrounds
## Aggregating results
## Selecting tuning parameters
## Fitting nrounds = 100, max_depth = 4, eta = 0.2, gamma = 0, colsample_bytree = 1, min_child_weight =
```

```r
# Make predictions on the training data
xgb_predicted_train <- predict(boosting_rf_model, newdata = data_train)
```

```r
# Compute confusion matrix and accuracy for training data
(xgb_confusion_matrix_train <- table(xgb_predicted_train, data_train$Survived))
```

```
##
## xgb_predicted_train   0   1
##                   0 360  35
##                   1  10 210
```

```r
(xgb_accuracy_train <- sum(diag(xgb_confusion_matrix_train)) /
    sum(xgb_confusion_matrix_train))
```

```
## [1] 0.9268293
```

```r
(xgb_misclassification_error_train <- 1 - xgb_accuracy_train)
```

```
## [1] 0.07317073
```

```r
# Make predictions on the test data
xgb_predicted_test <- predict(boosting_rf_model, newdata = data_test)

# Compute confusion matrix and accuracy for test data
(xgb_confusion_matrix_test <- table(xgb_predicted_test, data_test$Survived))
```

```
##
## xgb_predicted_test   0   1
##                   0 160  22
##                   1  15  75
```

```r
(xgb_accuracy_test <- sum(diag(xgb_confusion_matrix_test)) /
    sum(xgb_confusion_matrix_test))
```

```
## [1] 0.8639706
```

```r
(xgb_misclassification_error_test <- 1 - xgb_accuracy_test)
```

```
## [1] 0.1360294
```

```r
#create the comparison table based on the test data

# Create the data frame
comparison <- data.frame(
  model = c("blrc_testing", "knn_testing",
            "ann_testing", "nb_test", "svm_test",
            "dt_test", "dt_bag_test", "rf_test",
            "trf_test", "xgb_test"),
  accuracy = c(
              accuracy_blrc_testing,
              accuracy_knn_testing,
```

```
              ann_accuracy_testing,
              nb_accuracy_test,
              svm_accuracy_test,
              dt_accuracy_test,
              dt_bag_accuracy_test,
              rf_accuracy_test,
              trf_accuracy_test,
              xgb_accuracy_test),
  error = c(error_blrc_testing,
            misclassification_error_knn_testing,
            ann_misclassification_error_testing,
            nb_misclassification_error_test,
            svm_misclassification_error_test,
            dt_misclassification_error_test,
            dt_bag_misclassification_error_test,
            rf_misclassification_error_test,
            trf_misclassification_error_test,
            xgb_misclassification_error_test)
)

# Print the data frame
comparison
```

```
##           model  accuracy     error
## 1  blrc_testing 0.8125000 0.1875000
## 2   knn_testing 0.8272059 0.1727941
## 3   ann_testing 0.8198529 0.1801471
## 4       nb_test 0.8272059 0.1727941
## 5      svm_test 0.8455882 0.1544118
## 6       dt_test 0.8382353 0.1617647
## 7   dt_bag_test 0.8345588 0.1654412
## 8       rf_test 0.8566176 0.1433824
## 9      trf_test 0.8198529 0.1801471
## 10     xgb_test 0.8639706 0.1360294
```

```
# Find the row index of the model with the highest accuracy
best_model_index <- which.max(comparison$accuracy)

# Get the details of the best model
best_model <- comparison[best_model_index, ]

# Print the best model
print(best_model)
```

```
##       model  accuracy     error
## 10 xgb_test 0.8639706 0.1360294
```

```
#Write a reflection on your own word focusing on
#"what did I learn from this assignment?"
#From this assignment we learn to fit the different type of classifer with the
# given dataset. we split the data set and fit the various type of machine
#learning model that is generally used in the classification problem.
```

```
#we also calculate the confusion matrix and accuracy and error for each model.
#then we compare the accuracy of the test data set with different model and
#finally determine the best model.

#In our case the best model is the random forest boosting model with
#xgb boosting which gives the accuracy of 86% in the test data.
```