R Notebook

Libraries for the Ensemble Methods

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.2.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 4.2.3
```

```
library(mltools)
```

```
## Warning: package 'mltools' was built under R version 4.2.3
```

```
library(mccr)
```

```
## Warning: package 'mccr' was built under R version 4.2.3
```

Loading the data set.

```
# Reading in the data set.
setwd("C:/Users/amark/Downloads")
df <- read.csv("airlines.csv")

# Condensing the data down into a medium-sized set.
num1 <- sample(1:nrow(df), 0.98 * nrow(df), replace = FALSE)
df <- df[-num1,]
```

Bit of Data Exploration

```
# Converting DayOfWeek & Class (target) into factor variables.
df$DayOfWeek <- factor(df$DayOfWeek)
df$Class <- factor(df$Class)


# Splitting into 75% train and 25% test.
set.seed(1234) # For reproducible results.
num1 <- sample(nrow(df), 0.75 * nrow(df), replace = FALSE)
train <- df[num1,]
test <- df[-num1,]
```

Implementing RandomForest

```
# This is the randomForest method.
set.seed(1234) # For reproducible results.
print(rf <- randomForest(Class ~ ., data = train, importance = TRUE))
```

```
##
## Call:
```

```
##  randomForest(formula = Class ~ ., data = train, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 37.41%
## Confusion matrix:
##      0    1 class.error
## 0 3452 1129   0.2464527
## 1 1898 1612   0.5407407
```

```
pred1 <- predict(rf, newdata = test, type = "response")
print(acc_rf <- mean(pred1 == test$Class))
```

```
## [1] 0.6318131
```

```
print(mcc_rf <- mcc(factor(pred1), test$Class))
```

```
## [1] 0.2298092
```

Implementing XGBoost

```
# This is the XGBoost method.
set.seed(1234) # For reproducible results.

train_label <- ifelse(train$Class == 1, 1, 0)
train_matrix <- data.matrix(train[, -29])
model <- xgboost(data = train_matrix, label = train_label, nrounds = 100, objective = 'binary:logistic'
```

```
## [1]  train-logloss:0.437698
## [2]  train-logloss:0.296539
## [3]  train-logloss:0.207574
## [4]  train-logloss:0.148047
## [5]  train-logloss:0.106849
## [6]  train-logloss:0.077732
## [7]  train-logloss:0.056868
## [8]  train-logloss:0.041774
## [9]  train-logloss:0.030783
## [10] train-logloss:0.022742
## [11] train-logloss:0.016839
## [12] train-logloss:0.012494
## [13] train-logloss:0.009292
## [14] train-logloss:0.006927
## [15] train-logloss:0.005180
## [16] train-logloss:0.003887
## [17] train-logloss:0.002930
## [18] train-logloss:0.002221
## [19] train-logloss:0.001695
## [20] train-logloss:0.001304
## [21] train-logloss:0.001013
## [22] train-logloss:0.000796
## [23] train-logloss:0.000633
## [24] train-logloss:0.000510
## [25] train-logloss:0.000417
## [26] train-logloss:0.000345
## [27] train-logloss:0.000290
## [28] train-logloss:0.000246
```

```
## [29] train-logloss:0.000246
## [30] train-logloss:0.000246
## [31] train-logloss:0.000246
## [32] train-logloss:0.000246
## [33] train-logloss:0.000246
## [34] train-logloss:0.000246
## [35] train-logloss:0.000246
## [36] train-logloss:0.000246
## [37] train-logloss:0.000246
## [38] train-logloss:0.000246
## [39] train-logloss:0.000246
## [40] train-logloss:0.000246
## [41] train-logloss:0.000246
## [42] train-logloss:0.000246
## [43] train-logloss:0.000246
## [44] train-logloss:0.000246
## [45] train-logloss:0.000246
## [46] train-logloss:0.000246
## [47] train-logloss:0.000246
## [48] train-logloss:0.000246
## [49] train-logloss:0.000246
## [50] train-logloss:0.000246
## [51] train-logloss:0.000246
## [52] train-logloss:0.000246
## [53] train-logloss:0.000246
## [54] train-logloss:0.000246
## [55] train-logloss:0.000246
## [56] train-logloss:0.000246
## [57] train-logloss:0.000246
## [58] train-logloss:0.000246
## [59] train-logloss:0.000246
## [60] train-logloss:0.000246
## [61] train-logloss:0.000246
## [62] train-logloss:0.000246
## [63] train-logloss:0.000246
## [64] train-logloss:0.000246
## [65] train-logloss:0.000246
## [66] train-logloss:0.000246
## [67] train-logloss:0.000246
## [68] train-logloss:0.000246
## [69] train-logloss:0.000246
## [70] train-logloss:0.000246
## [71] train-logloss:0.000246
## [72] train-logloss:0.000246
## [73] train-logloss:0.000246
## [74] train-logloss:0.000246
## [75] train-logloss:0.000246
## [76] train-logloss:0.000246
## [77] train-logloss:0.000246
## [78] train-logloss:0.000246
## [79] train-logloss:0.000246
## [80] train-logloss:0.000246
## [81] train-logloss:0.000246
## [82] train-logloss:0.000246
```

```
## [83] train-logloss:0.000246
## [84] train-logloss:0.000246
## [85] train-logloss:0.000246
## [86] train-logloss:0.000246
## [87] train-logloss:0.000246
## [88] train-logloss:0.000246
## [89] train-logloss:0.000246
## [90] train-logloss:0.000246
## [91] train-logloss:0.000246
## [92] train-logloss:0.000246
## [93] train-logloss:0.000246
## [94] train-logloss:0.000246
## [95] train-logloss:0.000246
## [96] train-logloss:0.000246
## [97] train-logloss:0.000246
## [98] train-logloss:0.000246
## [99] train-logloss:0.000246
## [100]    train-logloss:0.000246
```

```
test_label <- ifelse(test$Class == 1, 1, 0)
test_matrix <- data.matrix(test[, -23])

probs <- predict(model, test_matrix)
pred2 <- ifelse(probs > 0.5, 1, 0)

print(acc_xg <- mean(pred2 == test_label))
```

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

```
print(mcc_xg <- mcc(pred2, test_label))
```

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

Summary: The ensemble method of randomForest ran at a moderate pace, given how large the data set truly is, even after condensing it to about 10,000 rows. Although the accuracies for the method was only around 54%, I would say the algorithm was less efficient compared to the faster and apparently accurate XGBoost, which its accuracy was 100% and I got quicker results than even logistic regression and naive baes, when those classification methods were ran.