Practical Machine Learning: Course Project

AAbellon1

2023-08-03

Executive Summary

Using fitness devices such as Fitbit, it is now possible to collect a large amount of data about personal activity relatively inexpensively. These types of devices enable individuals to take measurements about themselves to improve their health and to find patterns in their behavior. Interestingly, one thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. As such, the goal of this project is to use data from accelerometers on the belt, forearm, arm, and dumbbell of 6 participants and predict the manner in which they did the exercise. This is represented by the *classe* variable, consisting of 5 different ways or levels, found in the training dataset.

For this project, various machine learning models were created and compared to determine their relative effectiveness. Notably, the performance of Adaboost (a Boosting approach) and Random Forest using various K parameter values were examined, all implemented through the `RWeka` package. Overall, the Random Forest model with a K value of 2*sqrt(p) emerged as the most accurate, boasting a remarkable 99.7757% accuracy rate alongside a notably low anticipated out-of-sample error rate of 0.2243%.

Loading the Dataset and Libraries

First, the data and different libraries needed for the machine learning modeling were loaded.

```
fileUrl1 = 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
download.file(fileUrl1, destfile = './pml-training.csv',method = 'curl')

fileUrl2 = 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
download.file(fileUrl2, destfile = './pml-testing.csv',method = 'curl')

pml_training = read.csv("./pml-training.csv")
pml_testing = read.csv("./pml-testing.csv")

library(caret)
library(RWeka)
```

Data Preprocessing

Next, data preprocessing was conducted to account for the many NA and missing values found in the dataset. Cross validation was also done by using the `createDataPartition` function to split the data into training and test sets.

```
outTrain1 = sapply(pml_training, function(x) any(is.na(x)))
outTrain2 = sapply(pml_training, function(x) any("" %in% unique(x)))
outTrainFinal = outTrain1 | outTrain2
pml_training_clean = pml_training[,-which(outTrainFinal)]

set.seed(123)
inTrain = createDataPartition(pml_training_clean$classe, p=0.75, list=FALSE)
training = pml_training_clean[inTrain,]
testing = pml_training_clean[-inTrain,]
training = training[,-1:-7]
testing = testing[,-1:-7]
training[,1:(ncol(training)-1)] = lapply(training[,1:(ncol(training)-1)],as.numeric)
testing[,1:(ncol(testing)-1)] = lapply(testing[,1:(ncol(testing)-1)],as.numeric)
training$classe = as.factor(training$classe)
testing$classe = as.factor(testing$classe)
```

Model Selection

For the model selection, Adaboost and Random Forest (with different K parameter values) were created and evaluated through the following lines of code.

Adaboost (Boosting)

```
adaboosttree = AdaBoostM1(classe ~ ., data = training,
                          control = Weka_control(W=list(J48)))
evaluate_Weka_classifier(adaboosttree, newdata = testing)
```

```
##
## === Summary ===
##
## Correctly Classified Instances        4877               99.4494 %
## Incorrectly Classified Instances        27                0.5506 %
## Kappa statistic                          0.993
## Mean absolute error                      0.0023
## Root mean squared error                  0.045
## Relative absolute error                  0.7198 %
## Root relative squared error             11.3072 %
## Total Number of Instances             4904
##
## === Confusion Matrix ===
##
##     a    b    c    d    e   <-- classified as
##  1390    5    0    0    0 |    a = A
##     1  947    1    0    0 |    b = B
##     0    9  843    3    0 |    c = C
##     0    0    5  799    0 |    d = D
##     0    0    0    3  898 |    e = E
```

Random Forest (K = 0.5*sqrt(p))

```
RF = make_Weka_classifier('weka/classifiers/trees/RandomForest')
rfmodel1 = RF(classe ~ .,
              data = training,
              control = Weka_control(K=floor(0.5*sqrt(ncol(training)-1))))
evaluate_Weka_classifier(rfmodel1,newdata = testing)
```

```
##
## === Summary ===
##
## Correctly Classified Instances        4879               99.4902 %
## Incorrectly Classified Instances        25                0.5098 %
## Kappa statistic                          0.9936
## Mean absolute error                      0.0455
## Root mean squared error                  0.0867
## Relative absolute error                 14.3899 %
## Root relative squared error             21.8027 %
## Total Number of Instances             4904
##
## === Confusion Matrix ===
##
##     a    b    c    d    e   <-- classified as
##  1394    1    0    0    0 |    a = A
##     0  947    2    0    0 |    b = B
##     0    7  848    0    0 |    c = C
##     0    0   13  791    0 |    d = D
##     0    0    0    2  899 |    e = E
```

Random Forest (K = sqrt(p))

```
rfmodel2 = RF(classe ~ .,
              data = training,
              control = Weka_control(K=floor(sqrt(ncol(training)-1))))
evaluate_Weka_classifier(rfmodel2,newdata = testing)
```

```
##
## === Summary ===
##
## Correctly Classified Instances        4888               99.6737 %
## Incorrectly Classified Instances        16                0.3263 %
## Kappa statistic                          0.9959
## Mean absolute error                      0.0317
## Root mean squared error                  0.0689
## Relative absolute error                 10.0117 %
## Root relative squared error             17.3249 %
## Total Number of Instances             4904
##
## === Confusion Matrix ===
##
```

```
##     a    b    c    d    e   <-- classified as
##  1395    0    0    0    0 |    a = A
##     0  948    1    0    0 |    b = B
##     0    8  846    1    0 |    c = C
##     0    0    5  799    0 |    d = D
##     0    0    0    1  900 |    e = E
```

Random Forest (K = 2*sqrt(p))

```
rfmodel3 = RF(classe ~ .,
              data = training,
              control = Weka_control(K=floor(2*sqrt(ncol(training)-1))))
evaluate_Weka_classifier(rfmodel3,newdata = testing)
```

```
##
## === Summary ===
##
## Correctly Classified Instances        4893               99.7757 %
## Incorrectly Classified Instances        11                0.2243 %
## Kappa statistic                          0.9972
## Mean absolute error                      0.0248
## Root mean squared error                  0.0607
## Relative absolute error                  7.8488 %
## Root relative squared error             15.2748 %
## Total Number of Instances             4904
##
## === Confusion Matrix ===
##
##     a    b    c    d    e   <-- classified as
##  1395    0    0    0    0 |    a = A
##     0  948    1    0    0 |    b = B
##     0    7  847    1    0 |    c = C
##     0    0    0  804    0 |    d = D
##     0    0    0    2  899 |    e = E
```

From the results, we can see that although the accuracy between each of these machine learning models is minimal (i.e., Adaboost and all of the Random Forest models generated at least 99% accuracy and less than 1% out-of-sample error), it is ultimately **Random Forest (K = 2*sqrt(p))** which produced the highest accuracy rate of **99.7757%** and lowest out-of-sample error rate of **0.2243%**.

Predictions

Given this, Random Forest (K = 2*sqrt(p)) was then used as the model for predicting the 20 test cases available in the test data.

```
training_columns = c(colnames(training))
pred_columns = training_columns[-(length(training_columns))]
prediction_data = pml_testing[,pred_columns]

prediction_data[,1:(ncol(prediction_data))] = lapply(prediction_data[,1:(ncol(prediction_data))],as.num
```

```r
prediction_data$classe = predict(rfmodel3,prediction_data)
prediction_data$classe
```

```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```