PracticalMachineLearning

Ed Spinella

Friday, January 16, 2015

Background

This project utilizes sample exercise data to construct a model that predicts the manner in which people complete an exercise. The training and testing sample datasets include data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants who were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har

The outcome variable is "classe", a factor variable with 5 levels. For this data set, participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different fashions:

- exactly according to the specification (Class A)
- throwing the elbows to the front (Class B)
- lifting the dumbbell only halfway (Class C)
- lowering the dumbbell only halfway (Class D)
- throwing the hips to the front (Class E)

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Prediction evaluations will be based on maximizing the accuracy and minimizing the out-of-sample error. Relevant variables(features)will be used for prediction.

Decision tree and random forest algorithms will be used to construct each candidate model; the model with the highest accuracy will be chosen as final model.

Load Libraries,Data,Set Seed, Clean Data

```r
library(ElemStatLearn)
library(caret)
library(rpart)
library(randomForest)
library(AppliedPredictiveModeling)
set.seed(1234)
# Loading the train dataset replacing all missing with "NA"
trainset <- read.csv(file = 'train.csv', na.strings = c('NA','#DIV/0!',''))
# Loading the test dataset replacing all missing with "NA"
testset <- read.csv(file = 'test.csv',   na.strings = c('NA','#DIV/0!',''))

# Delete columns with all missing values
trainset <-trainset[,colSums(is.na(trainset)) == 0]
```

```
testset <-testset[,colSums(is.na(testset)) == 0]

# Delete columns 1-7 as they are irrelevant(related to the time-series or are
not numeric) to current project
trainset <- trainset[,-c(1:7)]
testset <- testset[,-c(1:7)]

# View the row(observations) and columns(features) that remain
dim(trainset); dim(testset)

## [1] 19622    53

## [1] 20 53
```

The training data set contains 53 variables and 19622 obs. The testing data set contains 53 variables and 20 obs.

Splitting Data into Testing and Cross-Validation

We subset our modified training data set into "train" and "test", which allows us to assess model accuracy.

```
subsamples <- createDataPartition(y=trainset$classe, p=0.75, list=FALSE)
train <- trainset[subsamples, ]
test <- trainset[-subsamples, ]
dim(train); dim(test)

## [1] 14718    53

## [1] 4904   53
```

We build our prediction models with both the Random Forest and Decision Tree Algorithms

Prediction Model 1: Random Forest

```
library(e1071)
library(randomForest)

model1 <- randomForest(train$classe ~. , data= train, method="class")

# Predicting:
prediction1 <- predict(model1, test, type = "class")

# Test results on test data set:
confusionMatrix(prediction1, test$classe)

## Confusion Matrix and Statistics
##
##            Reference
## Prediction    A    B    C    D    E
##          A 1395    1    0    0    0
##          B    0  946   11    0    0
##          C    0    2  843    8    0
##          D    0    0    1  796    0
##          E    0    0    0    0  901
## 
## Overall Statistics
##                                          
##                Accuracy : 0.9953         
##                  95% CI : (0.993, 0.997)
##     No Information Rate : 0.2845         
##     P-Value [Acc > NIR] : < 2.2e-16      
##                                          
##                   Kappa : 0.9941         
##  Mcnemar's Test P-Value : NA             
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9968   0.9860   0.9900   1.0000
## Specificity            0.9997   0.9972   0.9975   0.9998   1.0000
## Pos Pred Value         0.9993   0.9885   0.9883   0.9987   1.0000
## Neg Pred Value         1.0000   0.9992   0.9970   0.9981   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1929   0.1719   0.1623   0.1837
## Detection Prevalence   0.2847   0.1951   0.1739   0.1625   0.1837
## Balanced Accuracy      0.9999   0.9970   0.9917   0.9949   1.0000
```

Prediction Model 2: Decision Tree

```r
model2 <- rpart(classe ~ ., data=train, method="class")

# Predicting:
prediction2 <- predict(model2, test, type = "class")

confusionMatrix(prediction2, test$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1235  157   16   50   20
##          B   55  568   73   80  102
##          C   44  125  690  118  116
##          D   41   64   50  508   38
##          E   20   35   26   48  625
## 
## Overall Statistics
##                                           
##                Accuracy : 0.7394          
##                  95% CI : (0.7269, 0.7516)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6697
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8853   0.5985   0.8070   0.6318   0.6937
## Specificity            0.9307   0.9216   0.9005   0.9529   0.9678
## Pos Pred Value         0.8356   0.6469   0.6313   0.7247   0.8289
## Neg Pred Value         0.9533   0.9054   0.9567   0.9296   0.9335
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2518   0.1158   0.1407   0.1036   0.1274
## Detection Prevalence   0.3014   0.1790   0.2229   0.1429   0.1538
## Balanced Accuracy      0.9080   0.7601   0.8537   0.7924   0.8307
```

Results/Decision

The Random Forest algorithm performs better than Decision Trees and is therefore the chosen model for submission. Accuracy for Random Forest model is 0.995 (95% CI: (0.993, 0.997)) compared to 0.739 (95% CI: (0.727, 0.752)) for Decision Tree model.

The Random Forest model is choosen. The accuracy of the model is 0.995. **The expected out-of-sample error is estimated at 0.005, or 0.5% determined by subtracting the prediction accuracy of (.995) from 1.**

The chosen model provides the following prediction when run against the 20 test datset cases.

```
predictfinal <- predict(model1, testset, type="class")
predictfinal
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```