Machine Learning Assignment

Hans-Peter Bakker

09 December 2016

Introduction

This project requires the building of a model to forecast five classes of movement.

The data includes various columns that appear to be features created by the authors based on various statistics measured according to defined time lapses. According to the article. . . the authors recommended using 17 features including. . . .Although this would have been the preferred option in designing a model, the test set that was provided did not allow this option.

Reading in the data

```
#library(read.table)
#train_raw <- read.table("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", na.str
#test_raw <- read.table("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", na.strin
#write.csv(train_raw, file = "train.raw.csv"); write.csv(test_raw, file = "test.raw.csv")
```

nb cite: http://groupware.les.inf.puc-rio.br/har

Creating raw data files from the supplied training and test sets

```
train_raw <- read.csv("train.raw.csv")
test_raw <- read.csv("test.raw.csv")
```

Cleaning up the raw data sets:

This included the definition of the outcome variable as a factor variable with five levels (A,B,C,D,E), the removal of the first seven columns as well as the columns containing the time laps features.

```
#turning the "classe" outcome variable into a factor
train_raw$classe <- factor(train_raw$classe)

#removing the feature (NA) and the first seven columns:
cut <- c(1:7, 13:37,51:60,70:84,88:102,104:113,126:140,142:151)
train_raw <- train_raw[,-cut]
test_raw <- test_raw[,-cut]
```

For cross validation create test and training sets and check dimensions

In order to be able to explore the out-of-sample error estimates a training (70%) and test set were created using the data partitioning function of R.

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
set.seed(123)
inTrain <- createDataPartition(train_raw$classe, p = 0.7, list = FALSE)
training <- train_raw[inTrain,]
testing <- train_raw[-inTrain,]
dim(training) ; dim(testing)
```

```
## [1] 13737    54
```

```
## [1] 5885   54
```

Model fitting

Various models were considered, but given the categorical outcome variable, the preferred options were random forest and linear discriminant analysis. Principal Components Analysis was used to pre-process the linear discriminant analysis.

```
#First try random forest
#modFit1 <- train(classe ~., data = training, method = "rf", prox = TRUE)
#modFit1$finalModel
#pred1 <- predict(modFit1, testing)
#confusionMatrix(pred1, testing$classe)
#pred1_test <- predict(modFit1, test_raw)

#Secondly try linear discriminant analysis
modFit2 <- train(classe ~., data = training, method = "lda", preProcess = "pca")
```

```
## Loading required package: MASS
```

```
pred2 <- predict(modFit2, testing)
confusionMatrix(pred2, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1088  244  272   62  126
##          B  126  501  103  177  184
##          C  157  206  546  159  138
##          D  230  112   73  456  140
##          E   73   76   32  110  494
##
## Overall Statistics
##
##                Accuracy : 0.5242
##                  95% CI : (0.5114, 0.537)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3971
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.6499  0.43986  0.53216  0.47303  0.45656
## Specificity            0.8328  0.87568  0.86417  0.88722  0.93941
```

```
## Pos Pred Value         0.6071  0.45921  0.45274  0.45104  0.62930
## Neg Pred Value         0.8568  0.86692  0.89741  0.89577  0.88471
## Prevalence             0.2845  0.19354  0.17434  0.16381  0.18386
## Detection Rate         0.1849  0.08513  0.09278  0.07749  0.08394
## Detection Prevalence   0.3045  0.18539  0.20493  0.17179  0.13339
## Balanced Accuracy      0.7414  0.65777  0.69817  0.68012  0.69799
```

```
pred2_test <- predict(modFit2, test_raw)
```

Conclusion

The random forest suggests an out of sample accuracy of 99% while the linear discriminant analysis suggests a 54% accuracy. The downside of the random forest approach is the slow processing speed. This can be improved by considering making adjustments in the training controls. Although this may improve the process speed, it will most likely also reduce the high degree of expected accuracy.

End