Machine learning project report

Billy Harris

21/06/2015

Introduction

The aim of this study is to build a model to predict whether an exercise (dumbbell curl) is being carried out using correct technique, based on sensor data. For a brief overview of the original research and discussion of the use of sensors see: http://groupware.les.inf.puc-rio.br/har. A more detailed discussion is available in: Ugulino, W., Cardador, D. *et al.* "Wearable Computing: Accelerometers' Data Classification of Body Postures and Movement" in: *Lecture Notes in Computer Science*, pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012.

Method

The data and necessary libraries were loaded.

```
library(caret); library(ggplot2); library(rattle)
setwd("/home/billy/Documents/CourseraDataScience/MachineLearning/Project/")
rawdata<-read.csv("pml-training.csv")
questionset<-read.csv("pml-testing.csv")
set.seed(864)
```

The data consist of 19,622 observations, with 160 variables. Of these, the first 7 are contextual information (unique ID, test subject, timestamp etc.) and the 160th variable ("classe") is the variable the model will predict on. Classe is a factor variable with 5 levels:
A - correct technique
B - throwing the elbows to the front
C - lifting the dumbbell only halfway
D - lowering the dumbbell only halfway
E - throwing the hips to the front

The "test set" available for download on the Coursera website contained 20 cases for use in a separate exercise, and contained no classe variable. As such, a decision was made to split the "training" data set provided into training and test sets for model building purposes. Factor variables were coerced to numeric, following examination of their values to ensure the appropriateness of this approach.
Many of the variables contained large numbers of NA values, which can cause problems with many model building approaches and could result in a model that requires data not available in the testing set. A decision was made to filter the variables included in the model to include only those for which there were no missing values - this decision could always be revisited in the event of no suitable model being found based on the selected variables.

```
for (i in 8:159) {
    rawdata[[i]]<-as.numeric(as.character(rawdata[[i]]))
    questionset[[i]]<-as.numeric(as.character(questionset[[i]]))
} ;rm(i)

inTrain<-createDataPartition(y=rawdata$classe, p=0.8, list=FALSE)
training<-rawdata[inTrain,]
testing<-rawdata[-inTrain,]

selectVar<-function() {
    trainvars<-vector();testvars<-vector();questionvars<-vector()
    for (i in 8:159) {
```

```
        if (sum(is.na(training[[i]]))==0) {
            trainvars<-append(x = trainvars,values = i)
        }
        if (sum(is.na(testing[[i]]))==0) {
            testvars<-append(testvars,i)
        }
        if (sum(is.na(questionset[[i]]))==0) {
            questionvars<-append(questionvars,i)
        }
    }
    a<-intersect (trainvars, testvars)
    intersect(a,questionvars)
}
useVars<-selectVar()
trainData<-training[,c(useVars,160)]
testData<-testing[,c(useVars,160)]
```

Models were then constructed using the caret package in R. An approach based on classification trees was chosen because this does not assume normally distributed data and because the outcome factor variable has five levels (and would not be suitable for, for example, logistic regression).
The first attempted model was a simple classification tree using the "rpart" method. This model, which is not reproduced here, was not successful, having an accuracy of 0.49.
This approach was rejected and a second model constructed using the "gbm" method. This is a boosted approach, generating multiple classification trees and using a weighted output as the predictor. Cross-validation in this approach is built into model development, with groups of trees built on a resampled subset of the data (in the model below, 25 bootstrap resamples were conducted).

```
boostFit<-train(classe~.,data=trainData,method="gbm",verbose=FALSE)
```

This model had much higher accuracy, and is presented below.

Results
The boosted model is shown below, with a summary of the most important predictor variables.

```
boostFit
```

```
## Stochastic Gradient Boosting
##
## 15699 samples
##    52 predictors
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 15699, 15699, 15699, 15699, 15699, 15699, ...
##
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa      Accuracy SD
##   1                   50      0.7519836  0.6855725  0.007053737
##   1                  100      0.8188395  0.7707513  0.005443488
##   1                  150      0.8518166  0.8125021  0.003867719
```

```
##   2                   50      0.8520911  0.8126195  0.005361115
##   2                  100      0.9039974  0.8785380  0.004854353
##   2                  150      0.9268596  0.9074820  0.004512232
##   3                   50      0.8914110  0.8625772  0.004676785
##   3                  100      0.9374752  0.9209170  0.004160927
##   3                  150      0.9570600  0.9457007  0.003516878
##   Kappa SD
##   0.008998716
##   0.006794173
##   0.004836622
##   0.006773091
##   0.006090261
##   0.005647173
##   0.005871983
##   0.005213003
##   0.004414048
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
##
## Tuning parameter 'n.minobsinnode' was held constant at a value of 10
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were n.trees = 150,
##  interaction.depth = 3, shrinkage = 0.1 and n.minobsinnode = 10.
```

```
head(varImp(boostFit)$importance,10)
```

```
##                     Overall
## roll_belt        100.000000
## pitch_belt        16.805695
## yaw_belt          38.497354
## total_accel_belt   0.000000
## gyros_belt_x       0.000000
## gyros_belt_y       2.032405
## gyros_belt_z      15.938327
## accel_belt_x       0.000000
## accel_belt_y       0.000000
## accel_belt_z       5.096755
```

A confusion matrix was generated for both the training and test sets. The test set confusion matrix is reproduced below.

```
boostPred<-predict(boostFit,newdata=training)
testPred<-predict(boostFit,newdata=testing[,1:159])
```

```
confusionBoost<-confusionMatrix(boostPred,training$classe)
confusionTest<-confusionMatrix(data = testPred,reference = testing$classe)
confusionTest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
```

```
##          A 1095   26    0    0    1
##          B   15  718   25    2    5
##          C    2   14  651   20    3
##          D    3    0    7  616   14
##          E    1    1    1    5  698
##
## Overall Statistics
##
##                Accuracy : 0.963
##                  95% CI : (0.9567, 0.9687)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9532
##  Mcnemar's Test P-Value : 0.002384
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9812   0.9460   0.9518   0.9580   0.9681
## Specificity            0.9904   0.9851   0.9880   0.9927   0.9975
## Pos Pred Value         0.9759   0.9386   0.9435   0.9625   0.9887
## Neg Pred Value         0.9925   0.9870   0.9898   0.9918   0.9929
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2791   0.1830   0.1659   0.1570   0.1779
## Detection Prevalence   0.2860   0.1950   0.1759   0.1631   0.1800
## Balanced Accuracy      0.9858   0.9656   0.9699   0.9753   0.9828
```

The confusion matrix shows that the boosted model has an overall accuracy of 0.963 against the test set, for an out of sample error 0.037. For classe = A (correct performance), the model has a sensitivity of 0.98 and a specificity of 0.99.

Conclusion

This study concludes that the sensor data provides a suitable basis for building a predictive model (boosted classification tree) capable of detecting good form and errors in form in execution of an exercise (dumbbell curl) with a high degree of accuracy.