Predicting Correctly Executed Barbell Lifts From Data Collected by Wearable Accelerometers

Nicole Goebel

September 3, 2014

Executive Summary

The field of Human Acivity Recognition (HAR) has emerged from the technology of wearable devices that collect information about the quantity and quality of particular activities. In this HAR analysis, data was used from a study where six participants were fitted with accelerometers on the belt, forearm, arm and a dumbell. Each participant was instructed to lift the dumbell in one correct and four incorrect manners. I use this data from the wearable devices in order to train and test a data set with a random forest machine learning algorithm to build a model to predict whether a participant lifts the dumbell correctly or incorrectly. The most important predictive features identified by the algorithm across all factors of the response variable `classe` in the reduced training set include `roll_belt`, `pitch_belt`, `pitch_forearm`, `magnet_dumbbell_y`, `magnet_dumbbell_z` and `yaw_belt`. A cross-validation accuracy of aproximately 99% was obtained for manual and random forest cross-validation over 27 trees built. Predictions for a test set were submitted and correctly predicted all test observations (accuracy of 100%).

1. Introduction

Human activity recognition research has traditionally focused on predicting the types of activities performed. In this study, the non-traditional but potentially useful focus is on predicting how well an activity, lifting a dumbbell, is performed.

Six young (20-28 years old), inexperienced weightlifters were fitted with accelerometers on their belt, forearm, arm and a dumbell and were instructed to lift the dumbell in one correct (`classe A`) and four incorrect (`classes B-E`) manners. The incorrect manners were defined by throwing the elbows to the front (`classe B`), lifting the dumbbell only halfway (`classe C`), lowering the dumbbell only halfway (`classe D`) and throwing the hips to the front (`classe E`) (Velloso et. a. 2013). I apply a random forest machine learning algorithm to a labelled data set from the wearable devices in order to build a model that can predict whether a participant lifts the dumbell correctly or incorrectly.

Read more at: http://groupware.les.inf.puc-rio.br/har#ixzz3CgAE6SVZ

2. Preprocessing Datasets, Exploratory Analysis, and Feature Selection

Training and test data sets were downloaded from the https sites shown in the following code:

```
# set working directory and download data sets
setwd("/Users/nicolegoebel/GitHub/DataScienceCoursera/PracticalMachineLearning")
if (!file.exists("./test.csv")){
  trainURL="https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
  testURL="https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
  train<-download.file(trainURL, destfile="train.csv", method="curl")
  test<-download.file(testURL, destfile="test.csv", method="curl")
  dateDowloaded<-date()
  dput(dateDowloaded, "dateDownloaded")
} else {message("csv files already exist, downloaded on ", dget("dateDownloaded"))}
```

```
## csv files already exist, downloaded on Sun Sep  7 16:14:08 2014
```

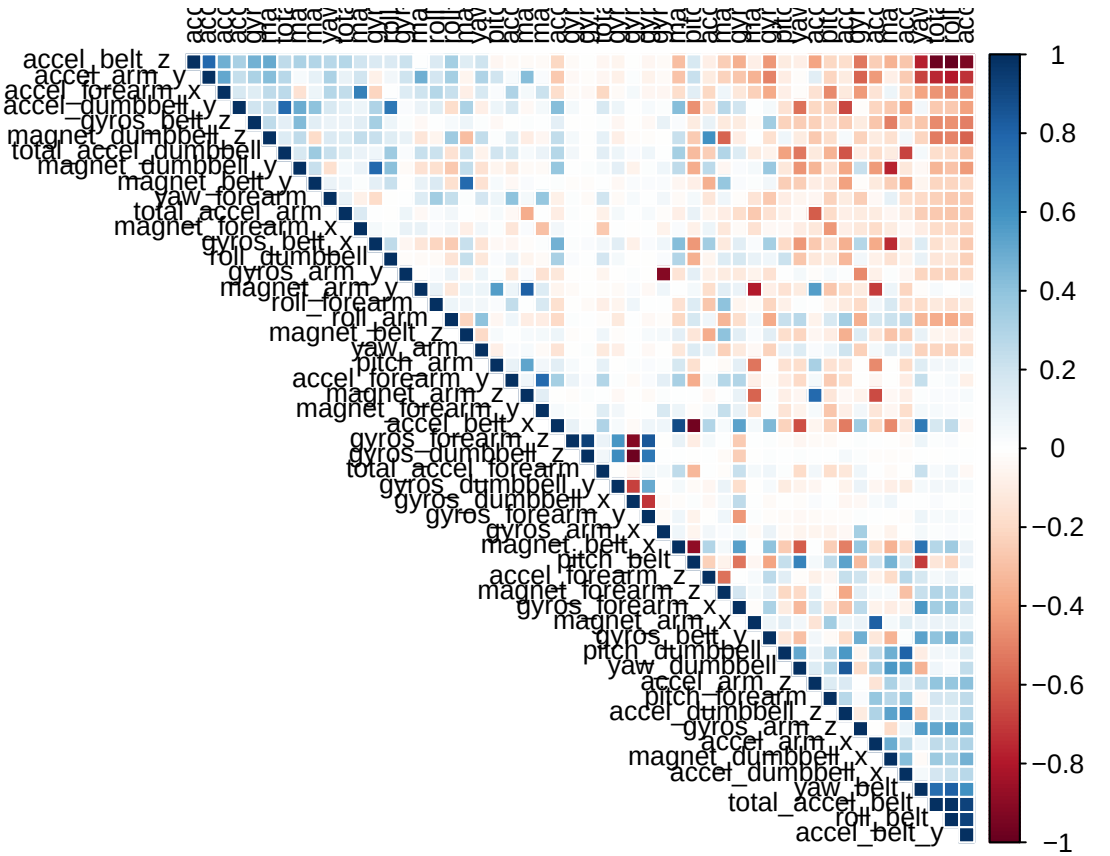
The csv files for the training and test datasets were loaded into R:

```
trainDataOrig <- read.csv("./train.csv", na.strings=c("NA","NaN", " ","DIV/0!"))#train set has 19622 rows x 160 co
testDataOrig <- read.csv("./test.csv", na.strings=c("NA","NaN", " ","DIV/0!")) #test set has 20 rows x 160 columns
headerTrain<-names(trainDataOrig)
headerTest<-names(testDataOrig)
```

Relevant variables (features) were retained in the training and testing datasets by removing unnecessary features (e.g., name, timestamps, etc) as well as those features with near zero variance as determined using nearZeroVar() function on the test set. Since a classification tree will be used to model the data, highly correlated variables were not removed, as classification trees are resistant to highly correlated

variables (Kuhn 2008). Out of curiosity, I calculated a correlation of features using instances where pairwise observations were complete in order to identify features with correlations greater than 90%. Figure 1 shows the correlations of the training set with reduced number of features.

```
library(ggplot2)
library(lattice)
library(corrplot)
library(caret)
#remove unnecessary features (e.g. name, timestamps, etc)
testData<-testDataOrig[,-(1:7)]
trainData<-trainDataOrig[,-(1:7)]
#remove near zero variance feature
nd1 <- dim(trainData)
nd1a=nd1[2]-7
nzvtest <- nearZeroVar(testData)
filteredTrain <- trainData[, -nzvtest]
filteredTest <- testData[, -nzvtest]
nd2 <- dim(filteredTrain)
featureCor<-cor(filteredTrain[,-nd2[2]], use="pairwise.complete.obs")
highCorr<-findCorrelation(featureCor,0.90)
numhighCorr<-length(highCorr)
#plot correlations
corrplot(featureCor, order = "FPC", method = "color", type = "upper", tl.cex = 0.8, tl.col = rgb(0, 0, 0))
```



data.pdf

Figure 1. Correlation matrix of features in the training data set, selected by non-near-zero variance features of the test set.

The training set included 19622 observations. The near zero variance filter reduced 146 numeric features to 53 features. Of these features, 7 were highly correlated at an $R^2 > 0.90$, but were retained in the data set for classification tree analysis.

The training set with reduced number of features was furher divided into a smaller training set (60%) and a validation test set (40%) for a manual cross-validation (in contrast to the cross-validation carried out by the random forest algorithm).

```
#subset filtered training data into a smaller training set and a validation set
inSet<-createDataPartition(filteredTrain$classe, p=0.6, list = FALSE)
trn<-filteredTrain[inSet,]
tst<-filteredTrain[-inSet,]
nd3 <- dim(trn)
```

3. Random Forest Model Build

A random Forest ensemble method was used to fit, predict, and aggregate many trees built, as well as identify the most important features. This method not only bootstraps the samples for each tree built, but also bootstraps the features at each split. Multiple trees are grown and the resulting prediction is based on an average over each tree's prediction. The random forest algorithm was performed on the training data set with a reduced number of features.

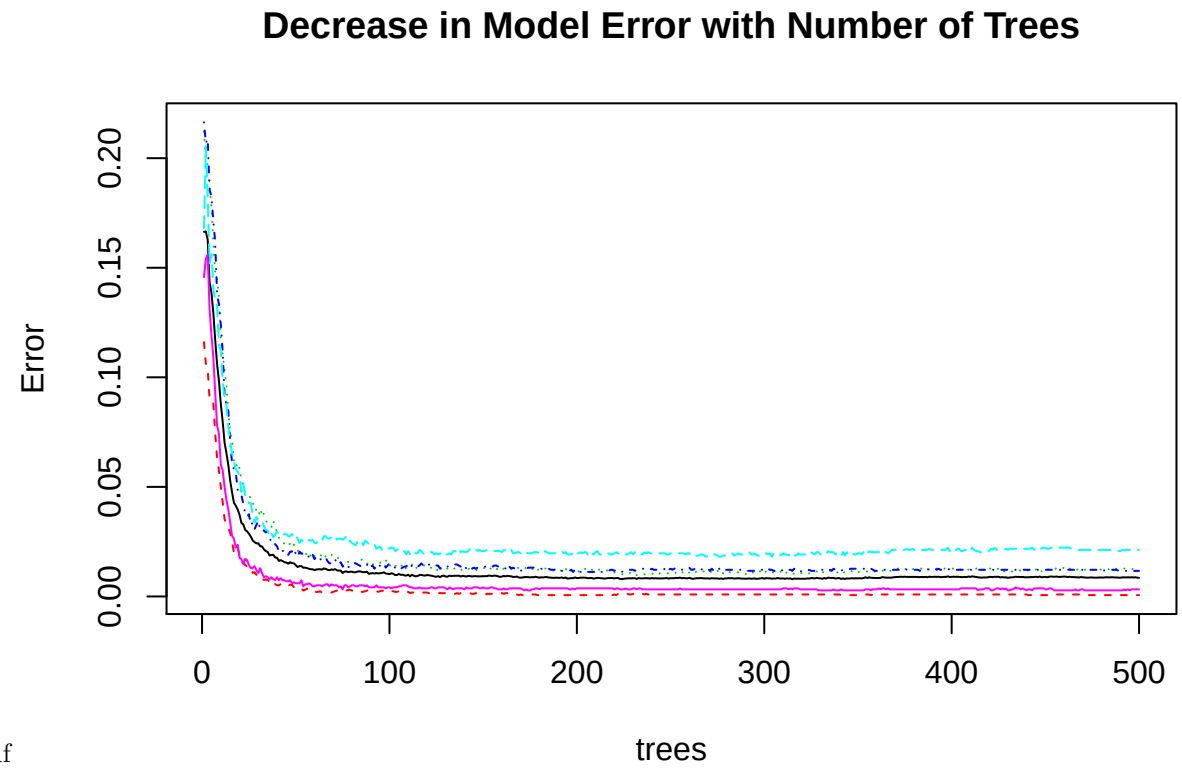
```
library(randomForest)
```

```
## randomForest 4.6-7
## Type rfNews() to see new features/changes/bug fixes.
```

```
forestFit <- train(classe~., data=trn, method="rf", importance=TRUE, trControl = trainControl(method = "cv", numbe
mtryBest<-forestFit$bestTune
mtryBestAccuracy<- max(forestFit$results[,2])
```

The random forest run on 53 features and 11776 observations took 2 mtrys to reach an accuracy of 0.9878 in a 4x cross validation. This is shown in Figure 2, which displays the decrease in model error with the number of trees built.

```
plot(forestFit$finalModel, main="Decrease in Model Error with Number of Trees")
```

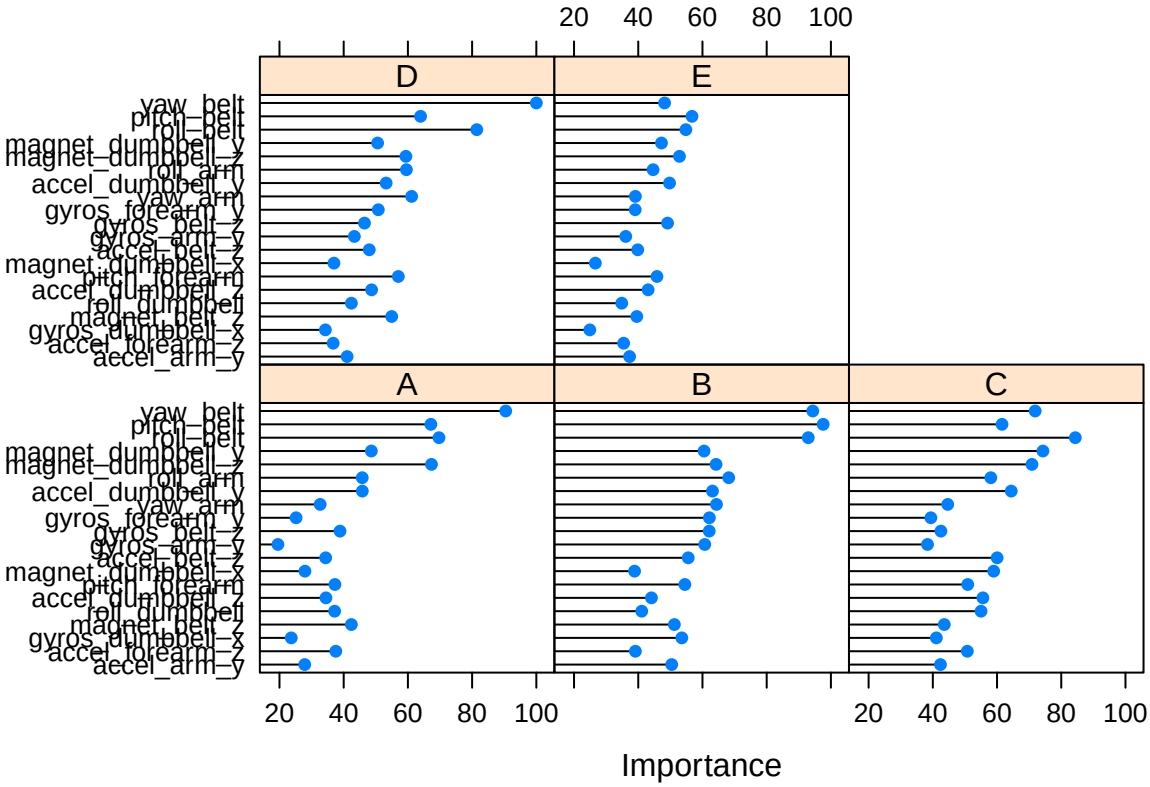


plot.pdf

Figure 2. Decrease in model error with the number of trees built using the random forest algorithm.

The importance of variables is shown in a Figure 3, which ranks the features from most to least important according to the role thy play in the random forest algorithm.

```
plot(varImp(forestFit), top = 20)
```



plot.pdf

Figure 3. Importance of the top 20 featurs (y-axis) assessed by the random forest model, ranked as a percentage from most to least important (x-axis).

The importance of features varied among the different response variable factors. The `classe` response factors demonstrate slightly different rankings of feature importance for the top 20 features (Fig. 3). For all `classe` factors, the most important features were `roll_belt`, `pitch_belt`, `pitch_forearm`, `magnet_dumbbell_y`, `magnet_dumbbell_z` and `yaw_belt`.

4. Cross-Validation: Model Accuracy and Out-of-Sample Error

Cross-validation and out of error estimates were calculated using the validation test set and the random forest algorithm function. Results from the prediction as based on the trained random forest model and the validation test set are shown below in a confusion matrix.

```
predVal<-predict(forestFit, tst[,-nd2[2]])
cm<-confusionMatrix(predVal,tst$classe)
cm$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 2229   13    0    0    0
##          B    3 1501   18    0    0
##          C    0    4 1348   30    1
##          D    0    0    2 1254    1
##          E    0    0    0    2 1440
```

```
acc<-cm$accur
modelAcc <- postResample(tst$classe, predVal)[[1]]
oose<-1-modelAcc
```

The confusion matrix from the cross-validation performed by the random forest function is also shown below:

```
forestFit$finalModel$confusion
```

```
##      A    B    C    D    E class.error
## A 3346    1    1    0    0   0.0005974
```

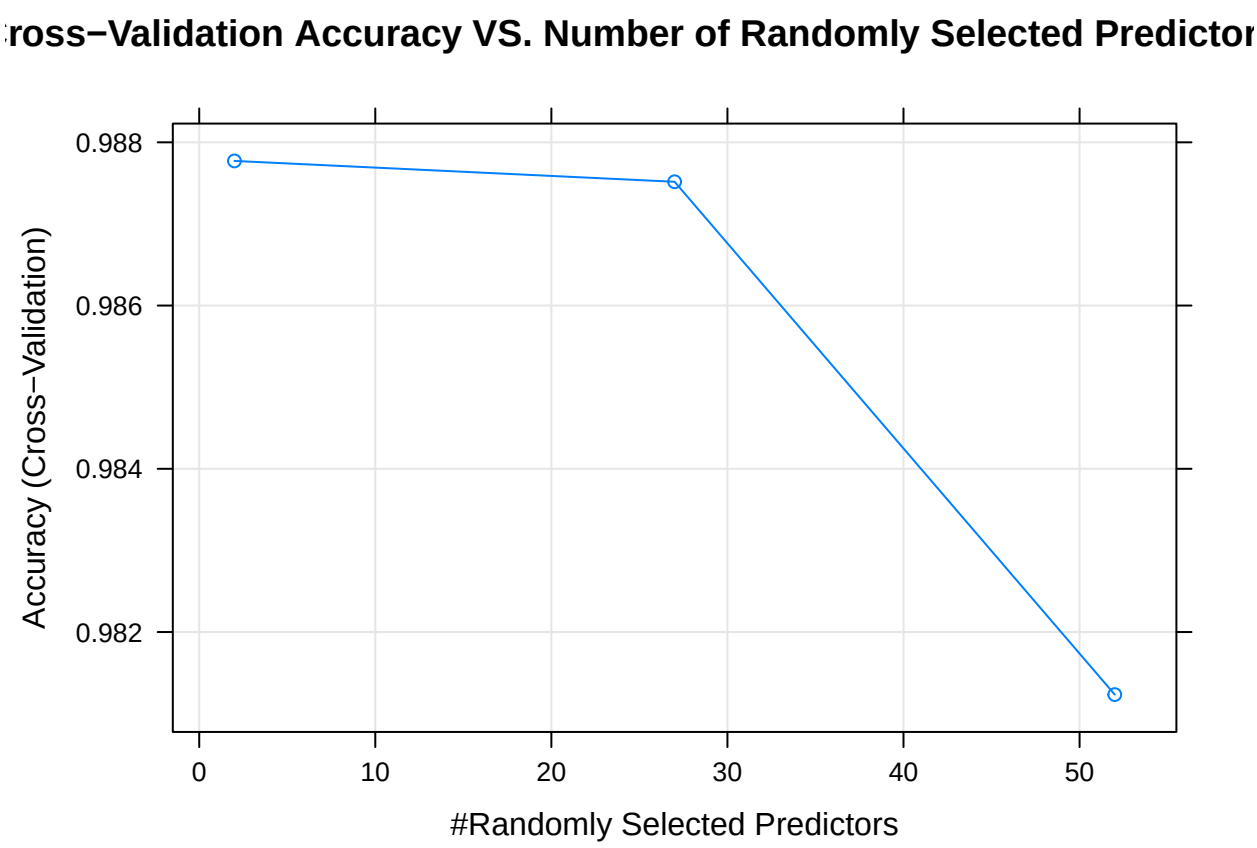
```
## B   17 2253    9    0    0   0.0114085
## C    1   20 2030    3    0   0.0116845
## D    0    0   39 1889    2   0.0212435
## E    0    0    2    5 2158   0.0032333
```

```
rs<-forestFit$resample
rfAvgAccuracy<-mean(rs[,1])
rfSdAccuracy<-sd(rs[,1])
rfOOSE<-1-rfAvgAccuracy
```

The confusion matrix from the manual cross validation shows the number of correct and incorrect predictions of the response variable `classe`. For example, 2229 predictions for `A` were correct, while 3 predictions were incorrect by predicting `B` instead of `A`. Based on this confusion matrix which compares the model predictions from the cross-validation datasets, an accuracy of 0.9906 and the corresponding out-of-sample-error (1-accuracy) of 0.0094 were calculated. These values were similar to the accuracy (mean=0.9878, stdev=0.0022) and corresponding out-of-sample-error (1-accuracy = 0.0122) calculated for the cross-validation carried out by the random forest algorithm. I expected a similar, or slightly lower, out-of-sample-error for the submitted predictions of the given test set (which did not supply the `classe` response variable (see below)).

Figure 4 demonstrates the change in cross-validaion accuracy with the number of randomly selected features in the random forest model. Average accuracy peaks at around 27 features and drops precipitously when all features are used.

```
plot(forestFit, main="Cross-Validation Accuracy VS. Number of Randomly Selected Predictors")
```



plot.pdf

Figure 4. Change in cross-validaion accuracy with the number of features randomly selected by the random forest algorithm.

5. Test Results

The formulated model was then applied to the features of test data set (with near zero variance features removed). The predictions from this test set were submitted for the assignment. Immediate feedback upon submission indicated that the model gave 100% accuracy. A higher accuracy of the test set than the training set suggests the possibility of the model overfitting.

```
forestAnswers<-predict(forestFit,filteredTest[,-nd2[2]])#, na.action="na.omit")
#function to write each prediction to a file for submission
pml_write_files = function(x){
  n = length(x)
```

```
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
#folder where filesare written.
mainDir<-"/Users/nicolegoebel/GitHub/DataScienceCoursera/PracticalMachineLearning/predictions"
modname<-"forestFit"
fp<-file.path(mainDir, modname)
dir.create(fp)
```

```
## Warning:
## '/Users/nicolegoebel/GitHub/DataScienceCoursera/PracticalMachineLearning/predictions/forestFit'
## already exists
```

```
setwd(fp)
pml_write_files(forestAnswers)
```

6. References

Kuhn, M. 2008. Building Predictive Models in R Using the caret Package. Journal of Statistical Software, Volume 28, Issue 5, p. 1.

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. 2013. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th Augmented Human (AH) International Conference in cooperation with ACM SIGCHI (Augmented Human'13) . Stuttgart, Germany: ACM SIGCHI.

Read more: http://groupware.les.inf.puc-rio.br/work.jsf?p1=11201#ixzz3CyZjSwu5