Practical Machine Learning Course Project - *how well they do it*

Konrad Mulrennan

Thursday, June 23, 2016

Synopsis

In this project, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har. The data collected refers to a Weight Lifting Exercises Dataset which has been published[1].

The target is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. Any of the other variables can be used to predict with. This report describes how the raw data was prepared, how the models were built and how eachj model was assessed. The resulting prediction model is chosen as the best to predict the class of the 20 different test cases.

Getting and Cleaning Data

The data is downloaded and stored in the train and test data sets. The train data set is further split into mytraining and mytesting data sets to train and test each of the models built. The mytraining data set will be used to build the models. The mytesting data set will be used to test the performance of each model against the others. The model which attains the best results will be used to predict the class of the exercises in the test data set.

The following lines of code are performed in the R studio console prior to compilation of this markdown document.

setwd("C:/Users/kmulrennan/Desktop/Data Science/Practical Machine Learning/Week 4")

download.file(url="https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", destfile = "train.csv")

download.file(url="https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", destfile = "test.csv")

```
#create the train data set from the training csv file
train <- read.csv("C:/Users/kmulrennan/Desktop/Data Science/Practical Machine Learning/Week 4/train.csv

#create the test data set from the testing csv file
test <- read.csv("C:/Users/kmulrennan/Desktop/Data Science/Practical Machine Learning/Week 4/test.csv")

#load the required libraries
library(AppliedPredictiveModeling)
library(caret)
library(randomForest)
library(ggplot2)
theme_set(theme_bw())

#split the training data set into further training (60%) and test (40%) sets
inTrain = createDataPartition(train$classe, p = 0.6, list = FALSE)
```

```
mytraining = train[inTrain,]
mytesting = train[-inTrain,]

#find the variables with near zero variance
myNZVvars <- nearZeroVar(mytraining, saveMetrics=TRUE)

#remove variables from the list which have zero variance
myNZVvars <- subset(myNZVvars, nzv == FALSE)

#create an ID column using the row names for the remaining variabes
myNZVvars$ID <- rownames(myNZVvars)

#only keep the ID column
myNZVvars <- myNZVvars[,5]

#subset the mytraining data frame keeping only the columns which names match those identified with
#having non zero variance
mytraining <- mytraining[, myNZVvars]

#remove columns from mytraining which consist of NA values which equate to greater than 50% of
#the observations
mytraining <- mytraining[, -which(colMeans(is.na(mytraining)) > 0.5)]

#create a list of column names which need to be dropped from the mytraining data frame
#these columns are to be dropped as they consist of Id and time stamp data which are not needed to
#predict how well the exercise is completed
drops <- c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_part_2", "cvtd_timestamp")

#this line of code drops the columns which are named in drops
mytraining <- mytraining[,!(names(mytraining) %in% drops)]

#creates a factor variable of the classe data in mytraining
train_classe <- mytraining$classe

#makes all columns bar the classe column numeric. The classe column is dropped from the data frame. A
#data matrix is created
mytraining <- lapply(mytraining[, -54], as.numeric)

#add the classe column back to the mytraining data set
mytraining$classe <- train_classe

#make mytraining a data frame
mytraining <- as.data.frame(mytraining)

#subset mytesting keeping only the same columns as mytraining
mytesting <- mytesting[, colnames(mytraining)]

#creates a factor variable of the classe data in mytesting
test_classe <- mytesting$classe

#makes all columns bar the classe column numeric. The classe column is dropped from the data frame. A
#data matrix is created
mytesting <- lapply(mytesting[, -54], as.numeric)
```

```
#add the classe column back to the mytesting data set
mytesting$classe <- test_classe

#make mytesting a data frame
mytesting <- as.data.frame(mytesting)

#creates a factor variable of the problem_id data in test
test_problem_id <- test$problem_id

#subset test keeping only the same columns as mytraining bar the classe column
test <- test[, colnames(mytraining[, -54])]

#make all of the columns numeric data type. This creates a data matrix
test <- lapply(test, as.numeric)

#make test a data frame
test <- as.data.frame(test)

#add the problem_id column back to the test data set
test$problem_id <- test_problem_id

#remove predictors from all data sets which have an absolute pairwise correlation > 0.9
cormytraining <- cor(mytraining[, -54])
highcor <- findCorrelation(cormytraining, 0.9)
mytraining <- mytraining[, -highcor]
mytesting <- mytesting[, -highcor]
test <- test[, -highcor]
```

Model building

To begin with the random number generator seed is set to ensure reproducibility of the results.Three different models are created using the mytraining data set. The models are boosted trees ("gbm"), random forest ("rf") and linear discriminant analysis ("lda"). Each model will then be used to predict the class of the exercises in the mytesting data set.A confusionMatrix will be performed on each of the model prediction results to assess each model. The model which performs the best will be chosen to predict the class of exercise in the test data set.

Boosted Trees Model

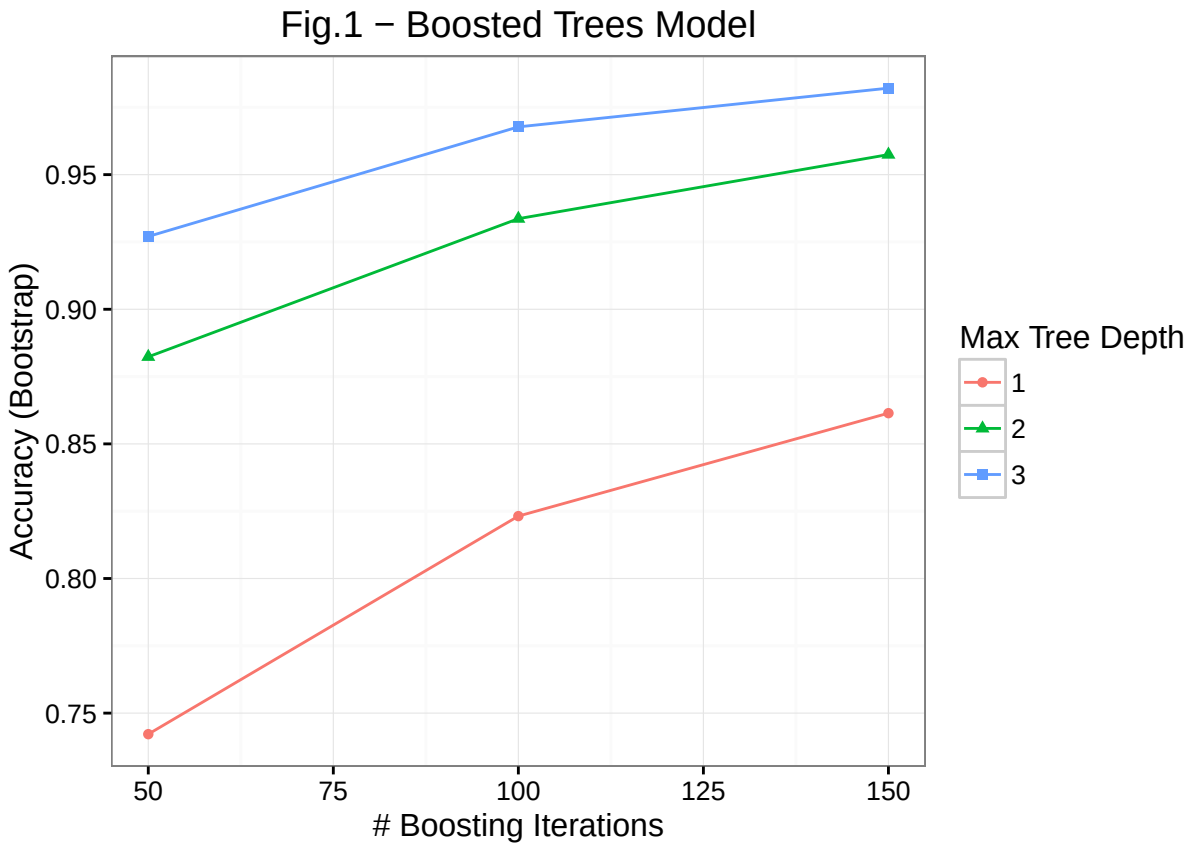
The Boosted Trees Model works quite well and returns an accuracy of 0.9864 or 98.64%. In the accuracy vs number of boosting iterations graph (*Fig.1*) it is the tree with a depth of 3 and 150 boosting iterations is chosen as the model. The tree depth refers to the number of nodes between the root node and the furthest leaf node.

```
#set the seed to 1010
set.seed(1010)

#mod1 is a boosted trees model
mod1 <- train(classe ~ ., method = "gbm", data = mytraining, verbose = FALSE)
```

```r
#pred1 uses mod1 to predict the class of exercise performed in the mytesting data set
pred1 <- predict(mod1, mytesting)

#this confusionMatrix assesses the performance of mod1 to correctly predict the class of exercise
#performed in the mytesting data set
confusionMatrix(mytesting$classe, pred1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2223    9    0    0    0
##          B   21 1468   26    3    0
##          C    0   10 1357    1    0
##          D    0    4   15 1267    0
##          E    1    3    5   17 1416
##
## Overall Statistics
##
##                Accuracy : 0.9853
##                  95% CI : (0.9824, 0.9879)
##     No Information Rate : 0.2861
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9815
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9902   0.9826   0.9672   0.9837   1.0000
## Specificity            0.9984   0.9921   0.9983   0.9971   0.9960
## Pos Pred Value         0.9960   0.9671   0.9920   0.9852   0.9820
## Neg Pred Value         0.9961   0.9959   0.9929   0.9968   1.0000
## Prevalence             0.2861   0.1904   0.1788   0.1642   0.1805
## Detection Rate         0.2833   0.1871   0.1730   0.1615   0.1805
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9943   0.9874   0.9828   0.9904   0.9980
```

```r
#the boosted trees model is plotted
ggplot(mod1) + ggtitle("Fig.1 - Boosted Trees Model")
```



Random Forest Model

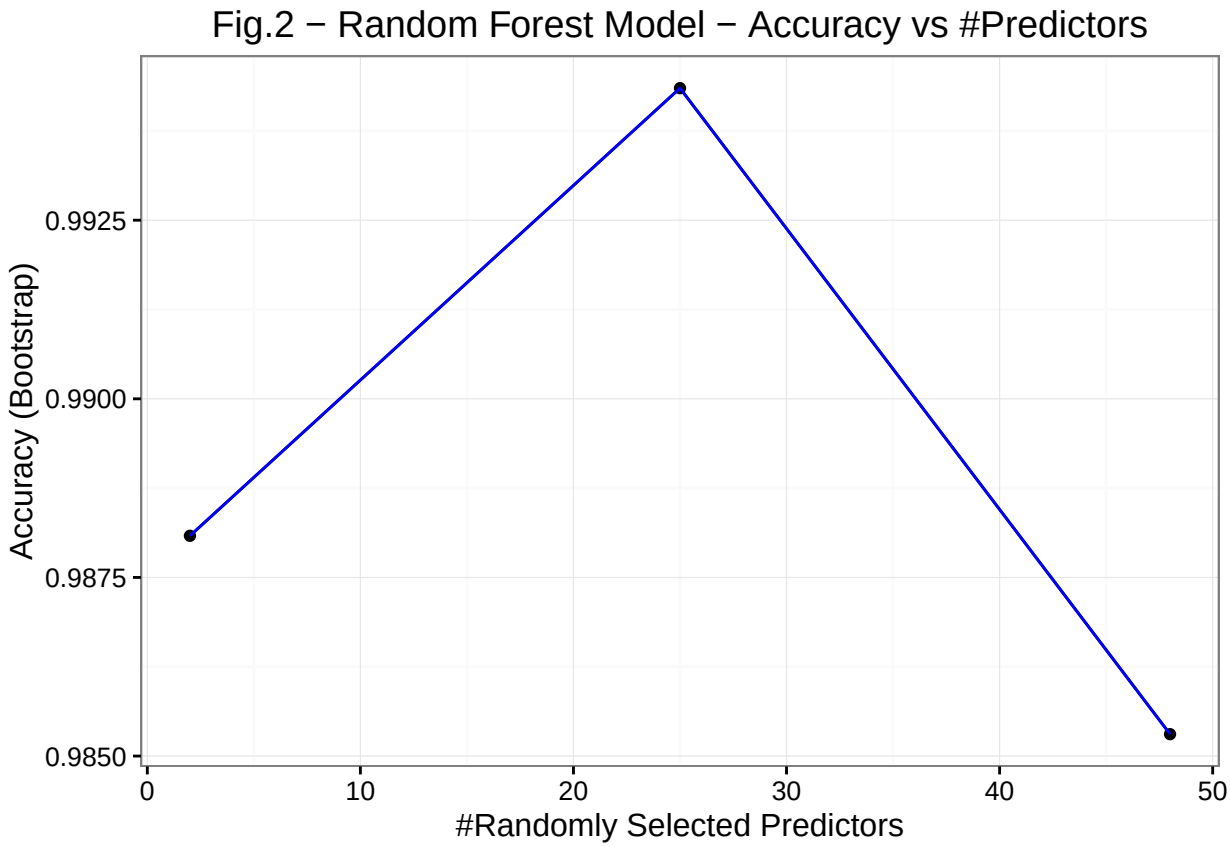
The cross validation graph (*Fig.2*) shows that the model with 27 predictors is selected as it has the best accuracy. Caret uses a cross validation method to select the number of predictors. The final model plot (*Fig.3*) communicates that the overall out of bag errors converge to zero at around 100 trees. This indicates that it is possible to speed up the algorithm by tuning the number of trees. The accuracy of the random forest model is 0.9949 indicating that the random forest model (mod2) is the best performing model with an accuracy of 99.49%.

```
#set the seed to 1010
set.seed(1010)

#mod2 is a random forest model
mod2 <- train(classe ~ ., method = "rf", data = mytraining)

#pred2 uses mod2 to predict the class of exercise performed in the mytesting data set
pred2 <- predict(mod2, mytesting)

#this confusionMatrix assesses the performance of mod2 to correctly predict the class of exercise
#performed in the mytesting data set
confusionMatrix(mytesting$classe, pred2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

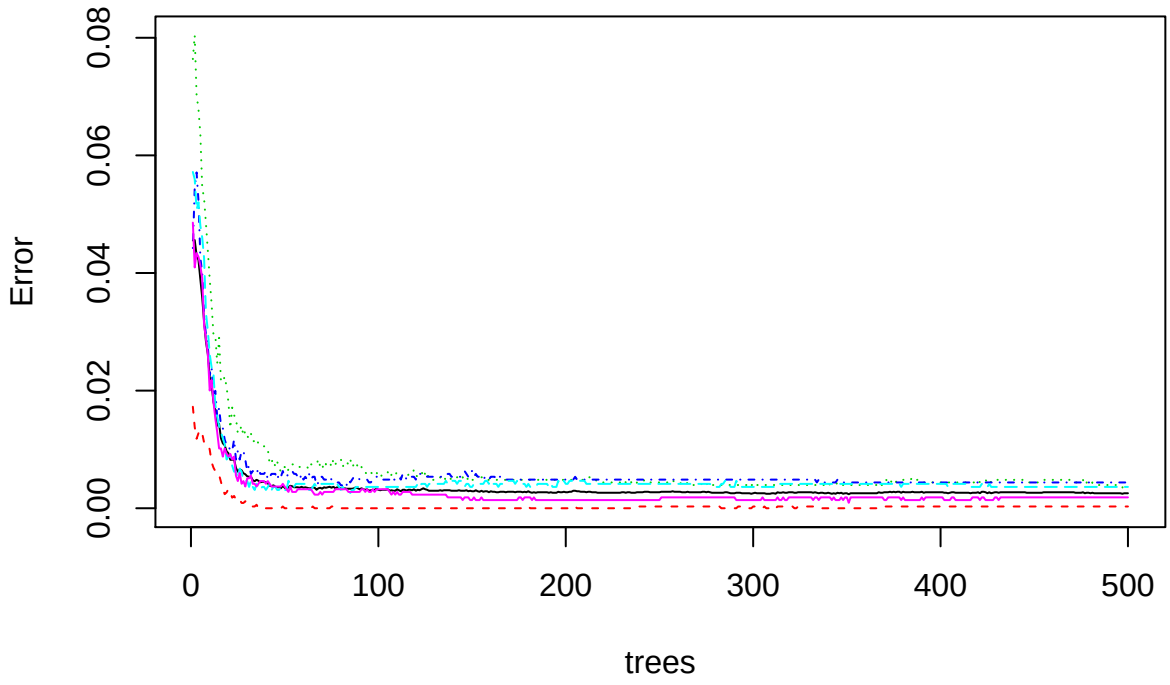
```
## Prediction    A    B    C    D    E
##          A 2230    1    0    0    1
##          B    3 1506    9    0    0
##          C    0    2 1366    0    0
##          D    0    0    3 1283    0
##          E    0    0    0    3 1439
##
## Overall Statistics
##
##                Accuracy : 0.9972
##                  95% CI : (0.9958, 0.9982)
##     No Information Rate : 0.2846
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9965
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9980   0.9913   0.9977   0.9993
## Specificity            0.9996   0.9981   0.9997   0.9995   0.9995
## Pos Pred Value         0.9991   0.9921   0.9985   0.9977   0.9979
## Neg Pred Value         0.9995   0.9995   0.9981   0.9995   0.9998
## Prevalence             0.2846   0.1923   0.1756   0.1639   0.1835
## Detection Rate         0.2842   0.1919   0.1741   0.1635   0.1834
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9992   0.9981   0.9955   0.9986   0.9994
```

```
#the random forest model is plotted
ggplot(mod2) +
        ggtitle("Fig.2 - Random Forest Model - Accuracy vs #Predictors") +
        geom_line(color="blue")
```




```
plot(mod2$finalModel, main="Fig.3 - Random Forest Model - OOB Error vs #Trees")
```

Fig.3 − Random Forest Model − OOB Error vs #Trees



Linear Discriminant Analysis Model

This model has the lowest accuracy 0.7165 or 71.65% and was not considered any further. This would indicate that there is not a strong linear relationship between variables which can best describe the data to predict the 'classe' outcome.

```
#set the seed to 1010
set.seed(1010)

#mod3 is a linear discriminant analysis model
mod3 <- train(classe ~ ., method = "lda", data = mytraining)

#pred3 uses mod3 to predict the class of exercise performed in the mytesting data set
pred3 <- predict(mod3, mytesting)

#this confusionMatrix assesses the performance of mod3 to correctly predict the class of exercise
#performed in the mytesting data set
confusionMatrix(mytesting$classe, pred3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1857   83  122  159   11
##          B  204  939  208   78   89
```

```
##          C  165  110  884  160   49
##          D   72   96  165  896   57
##          E   67  208  126  164  877
##
## Overall Statistics
##
##                Accuracy : 0.695
##                  95% CI : (0.6847, 0.7052)
##     No Information Rate : 0.3014
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6137
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7852   0.6539   0.5874   0.6150   0.8098
## Specificity            0.9316   0.9097   0.9237   0.9390   0.9165
## Pos Pred Value         0.8320   0.6186   0.6462   0.6967   0.6082
## Neg Pred Value         0.9095   0.9215   0.9041   0.9145   0.9678
## Prevalence             0.3014   0.1830   0.1918   0.1857   0.1380
## Detection Rate         0.2367   0.1197   0.1127   0.1142   0.1118
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.8584   0.7818   0.7555   0.7770   0.8631
```

Predictions

The Random Forest Model is now used to predict the exercise class of the test data set as it has the greatest accuracy of all 3 models.

```
#predtest uses mod2 to predict the class of exercise performed in the test data set
predtest <- predict(mod2, test)

#a data frame is created to store the results from predtest against the problem_id
result <- data.frame(test$problem_id, predtest)

#the results of the class of exercise predicted for the test data set are printed
result
```

```
##    test.problem_id predtest
## 1                1        B
## 2                2        A
## 3                3        B
## 4                4        A
## 5                5        A
## 6                6        E
## 7                7        D
## 8                8        B
## 9                9        A
## 10              10        A
## 11              11        B
```

```
## 12               12        C
## 13               13        B
## 14               14        A
## 15               15        E
## 16               16        E
## 17               17        A
## 18               18        B
## 19               19        B
## 20               20        B
```

References

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

Read more: http://groupware.les.inf.puc-rio.br/har#ixzz4CDmu1dAG