Practical Machine Learning Prediction Assignment

Lisa Post

July 6, 2018

Project Synopsis

Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be 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. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Goal

The goal of the project 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 model was built, how cross validation was used, what the expected out of sample error is, and why the choices were made. The prediction model will also be used to predict 20 different test cases.

Loading the Data

```
library(caret)
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.3.3
```

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 3.3.3
```

```
library(rpart.plot)
library(RColorBrewer)
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.3.3
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.3.3
```

```
set.seed(12345)

trainingUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testingUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

training <- read.csv(url(trainingUrl), na.strings=c("NA", "#DIV/0!", ""))
testing <- read.csv(url(testingUrl), na.strings=c("NA", "#DIV/0!", ""))
```

The training dataset is partitioned into two.

```
inTrain <- createDataPartition(training$classe, p=0.7, list=FALSE)
Training_Set <- training[inTrain, ]
Testing_Set <- training[-inTrain, ]
dim(Training_Set); dim(Testing_Set)
```

```
## [1] 13737   160
```

```
## [1] 5885  160
```

Cleaning the Data

The Near Zero Variance (NZV) variables are removed.

```
NZV <- nearZeroVar(Training_Set)
Training_Set <- Training_Set[, -NZV]
Testing_Set <- Testing_Set[, -NZV]
dim(Training_Set); dim(Testing_Set)
```

```
## [1] 13737   130
```

```
## [1] 5885  130
```

Variables that are mostly NA are removed.

```
All_NA <- sapply(Training_Set, function(x) mean(is.na(x))) > 0.95
Training_Set <- Training_Set[, All_NA==FALSE]
Testing_Set <- Testing_Set[, All_NA==FALSE]
dim(Training_Set); dim(Testing_Set)
```

```
## [1] 13737    59
```

```
## [1] 5885   59
```

Variables that are identification only and not useful for prediction are removed.

```
Training_Set <- Training_Set[, -(1:5)]
Testing_Set <- Testing_Set[, -(1:5)]
dim(Training_Set); dim(Testing_Set)
```

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

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

After cleaning the data, the number of variables for the analysis has been reduced to 54.

Prediction Model Building

Three models are applied to the training dataset: Decision Tree, Generalized Boosted Model and Random Forest. The accuracies of each model when applied to the testing dataset is analyzed.

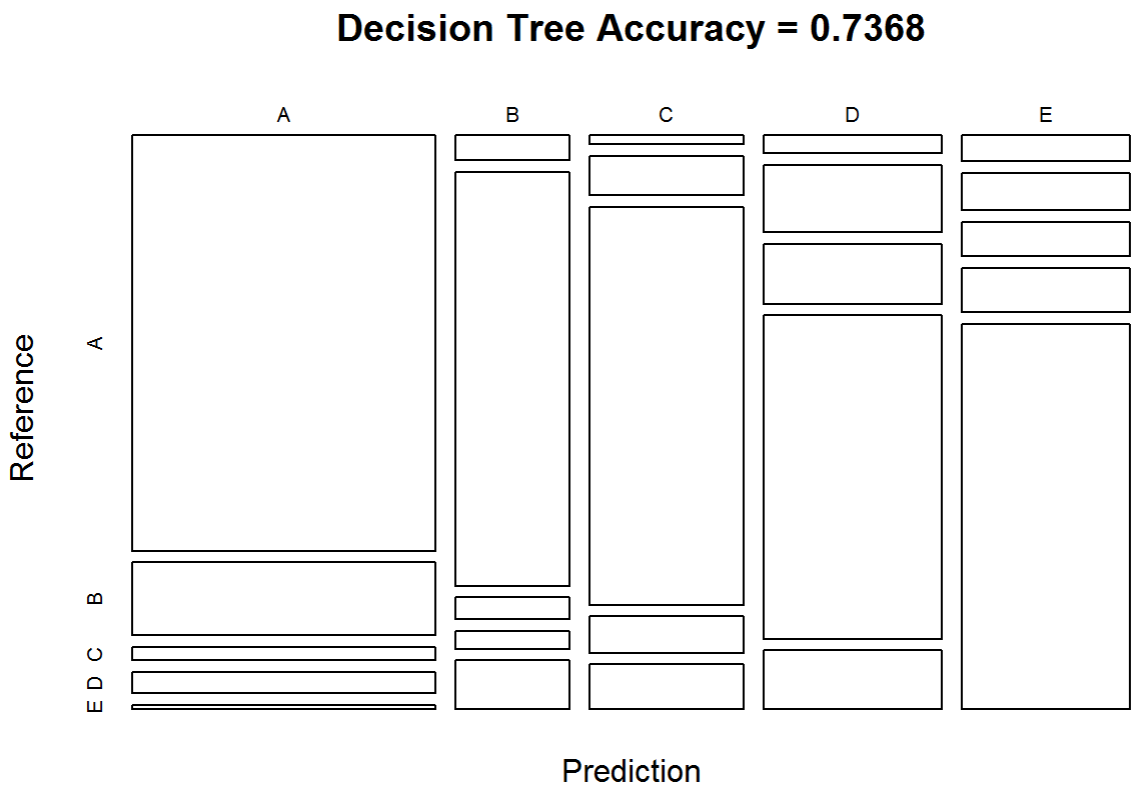
Decision Tree

```
set.seed(12345)
Mod_Fit_Dec_Tree <- rpart(classe ~ ., data=Training_Set, method="class")
Predict_Dec_Tree <- predict(Mod_Fit_Dec_Tree, newdata=Testing_Set, type="class")
Conf_Mat_Dec_Tree <- confusionMatrix(Predict_Dec_Tree, Testing_Set$classe)
```

```
Conf_Mat_Dec_Tree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1530  269   51   79   16
##          B   35  575   31   25   68
##          C   17   73  743   68   84
##          D   39  146  130  702  128
##          E   53   76   71   90  786
##
## Overall Statistics
##
##                Accuracy : 0.7368
##                  95% CI : (0.7253, 0.748)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6656
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9140  0.50483   0.7242   0.7282   0.7264
## Specificity            0.9014  0.96650   0.9502   0.9100   0.9396
## Pos Pred Value         0.7866  0.78338   0.7543   0.6131   0.7305
## Neg Pred Value         0.9635  0.89051   0.9422   0.9447   0.9384
## Prevalence             0.2845  0.19354   0.1743   0.1638   0.1839
## Detection Rate         0.2600  0.09771   0.1263   0.1193   0.1336
## Detection Prevalence   0.3305  0.12472   0.1674   0.1946   0.1828
## Balanced Accuracy      0.9077  0.73566   0.8372   0.8191   0.8330
```

```
plot(Conf_Mat_Dec_Tree$table, col = Conf_Mat_Dec_Tree$byClass, main = paste("Decision Tree Acc
uracy =", round(Conf_Mat_Dec_Tree$overall['Accuracy'], 4)))
```



Generalized Boosted Model

```
set.seed(12345)
Control_GBM <- trainControl(method="repeatedcv", number=5, repeats=1)
Mod_Fit_GBM <- train(classe ~ ., data=Training_Set, method="gbm", trControl=Control_GBM, verbo
se=FALSE)
Mod_Fit_GBM$finalModel
```

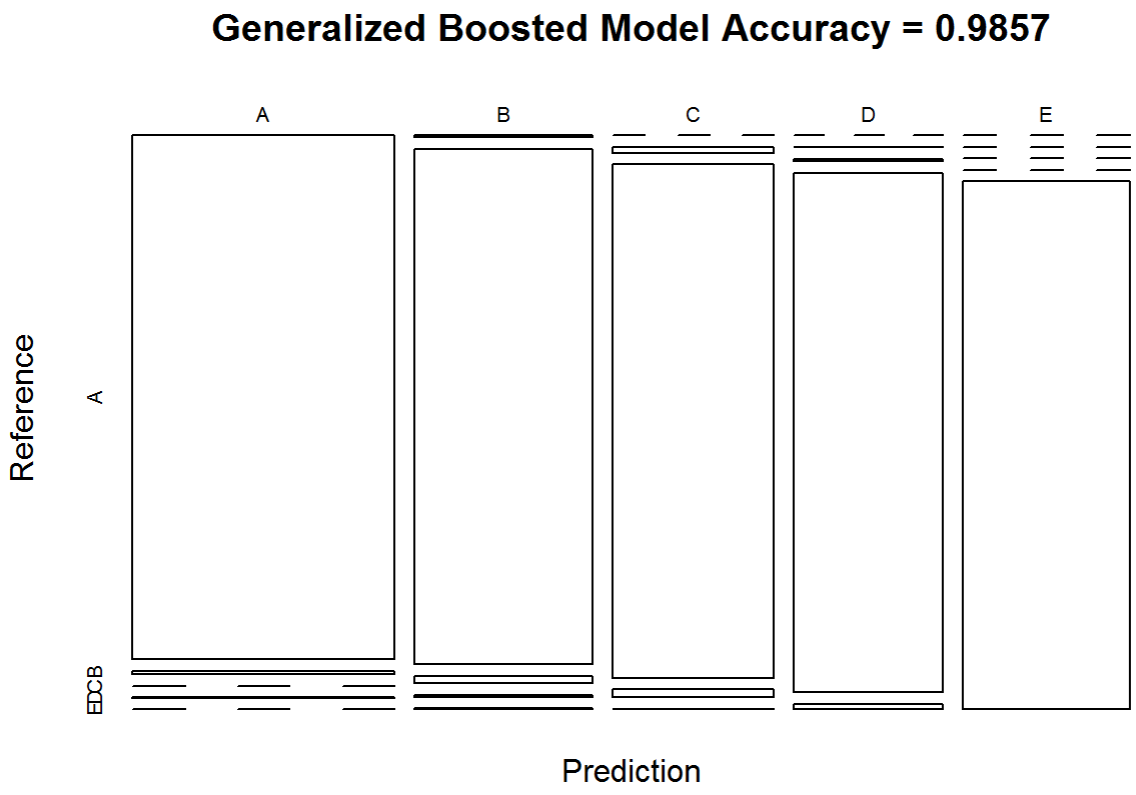
```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 41 had non-zero influence.
```

```
Predict_GBM <- predict(Mod_Fit_GBM, newdata=Testing_Set)
Conf_Mat_GBM <- confusionMatrix(Predict_GBM, Testing_Set$classe)
Conf_Mat_GBM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1670   11    0    2    0
##          B    4 1115   16    5    2
##          C    0   12 1006   16    1
```

```
##          D    0    1    4  941   10
##          E    0    0    0    0 1069
##
## Overall Statistics
##
##                Accuracy : 0.9857
##                  95% CI : (0.9824, 0.9886)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9819
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9789   0.9805   0.9761   0.9880
## Specificity            0.9969   0.9943   0.9940   0.9970   1.0000
## Pos Pred Value         0.9923   0.9764   0.9720   0.9843   1.0000
## Neg Pred Value         0.9990   0.9949   0.9959   0.9953   0.9973
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2838   0.1895   0.1709   0.1599   0.1816
## Detection Prevalence   0.2860   0.1941   0.1759   0.1624   0.1816
## Balanced Accuracy      0.9973   0.9866   0.9873   0.9865   0.9940
```

```
plot(Conf_Mat_GBM$table, col = Conf_Mat_GBM$byClass, main = paste("Generalized Boosted Model A
ccuracy =", round(Conf_Mat_GBM$overall['Accuracy'], 4)))
```



Random Forest

```
set.seed(12345)
Control_RF <- trainControl(method="cv", number=3, verboseIter=FALSE)
Mod_Fit_RF <- train(classe ~ ., data=Training_Set, method="rf", trControl=Control_RF)
Mod_Fit_RF$finalModel
```

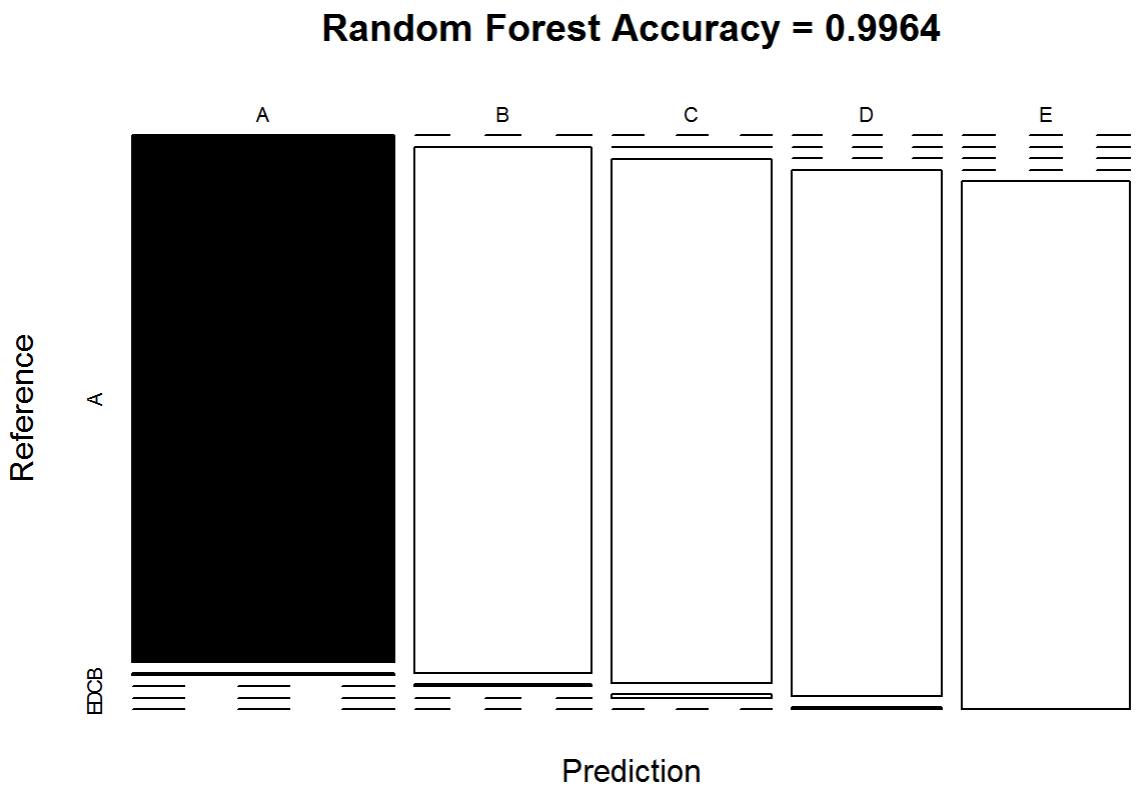
```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.2%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    1    0    0    1 0.0005120328
## B    6 2649    2    1    0 0.0033860045
## C    0    4 2391    1    0 0.0020868114
## D    0    0    7 2245    0 0.0031083481
## E    0    0    0    5 2520 0.0019801980
```

```
Predict_RF <- predict(Mod_Fit_RF, newdata=Testing_Set)
```

```
Conf_Mat_RF <- confusionMatrix(Predict_RF, Testing_Set$classe)
Conf_Mat_RF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    4    0    0
##          C    0    1 1022    7    0
##          D    0    0    0  957    4
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.9964
##                  95% CI : (0.9946, 0.9978)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9955
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9961   0.9927   0.9963
## Specificity            0.9988   0.9992   0.9984   0.9992   1.0000
## Pos Pred Value         0.9970   0.9965   0.9922   0.9958   1.0000
## Neg Pred Value         1.0000   0.9987   0.9992   0.9986   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1737   0.1626   0.1832
## Detection Prevalence   0.2853   0.1932   0.1750   0.1633   0.1832
## Balanced Accuracy      0.9994   0.9969   0.9972   0.9960   0.9982
```

```
plot(Conf_Mat_RF$table, col = Conf_Mat_RF$byClass, main = paste("Random Forest Accuracy =", ro
und(Conf_Mat_RF$overall['Accuracy'], 4)))
```



Model Selection and Predicting Results on the Test Data

The Random Forest model has the most accurate results at 99.64% with an expected out of sample error of 100 - 99.64 = 0.36%. The Random Forest model will be applied to predict the 20 quiz results from the original testing dataset.

```
Predict_Quiz <- predict(Mod_Fit_RF, newdata=testing)
Predict_Quiz
```

```
##  [1] 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
```