# Practical Machine Learning

*November 2015*

# Introduction

This report describes the analysis performed to create a machine learning algorithm to predict the classification assigned to each of 20 records representing the manner in which the weight lifting exercise was performed.

# Approach

Exploratory data analysis was peformed to determine which variables to use in the model. Non-numeric columns, except for the outcome column, classe, were discarded. Also discarded were numeric columns with NA values and those columns containing the record number, username, timestamps, and window information. This left a set of 51 predictors.

The training file provided in the assignment was split 70/30 into training and test datasets. The following iterative process was followed to identify the model with the best accurarcy.

1. Select model type and parameters.
2. Train the model using the training set.
3. Evaluate the model using the test set.
4. Repeat steps 2 - 4 with a different model type and/or parameters.

The Stochastic Gradient Boosting model type was used with a 10-fold cross-validation. Preprocessing activities centering and scaling were applied. Parameters that controlled tree complexity (interaction.depth) and number of iterations (n.trees) were adjusted, while the learning rate parameter (shrinkage) and minium number of training set samples in a node to commece splitting (m.minobsinnode) were left constant.

# Initial Model

The initial model produced a 0.9616387 accuracy rate on the training subset and a 0.9615973 accuracy rate on the test set. This corresponds to an out of sample error of 0.0383613 and 0.0384027, respectively.

```
fitControl <- trainControl(
        method = "cv",
        number = 10)

set.seed(myseed)
cl <- makeCluster(no_cores)
registerDoParallel(cl)
gbmFit1 <- train(classe ~ ., data = mytrain,
                 method = "gbm",
                 preProc = c('center', 'scale'),
                 trControl = fitControl,
                 verbose = FALSE)
stopCluster(cl)
gbmFit1
```

```
## Stochastic Gradient Boosting
##
## 13737 samples
##    51 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## Pre-processing: centered (51), scaled (51)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12361, 12364, 12363, 12364, 12364, 12365, ...
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa      Accuracy SD
##   1                   50      0.7556994  0.6900652  0.016299262
##   1                  100      0.8223061  0.7751366  0.010372097
##   1                  150      0.8546986  0.8161417  0.010428090
##   2                   50      0.8579755  0.8200593  0.011258193
##   2                  100      0.9066037  0.8818242  0.009348676
##   2                  150      0.9320079  0.9139824  0.007774130
##   3                   50      0.8961930  0.8686562  0.010553171
##   3                  100      0.9413271  0.9257640  0.007054096
##   3                  150      0.9614173  0.9511810  0.005258582
##   Kappa SD
##   0.020738656
##   0.013111649
##   0.013205104
##   0.014286270
##   0.011837785
##   0.009828838
##   0.013338693
##   0.008940485
##   0.006666411
##
## 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.
```

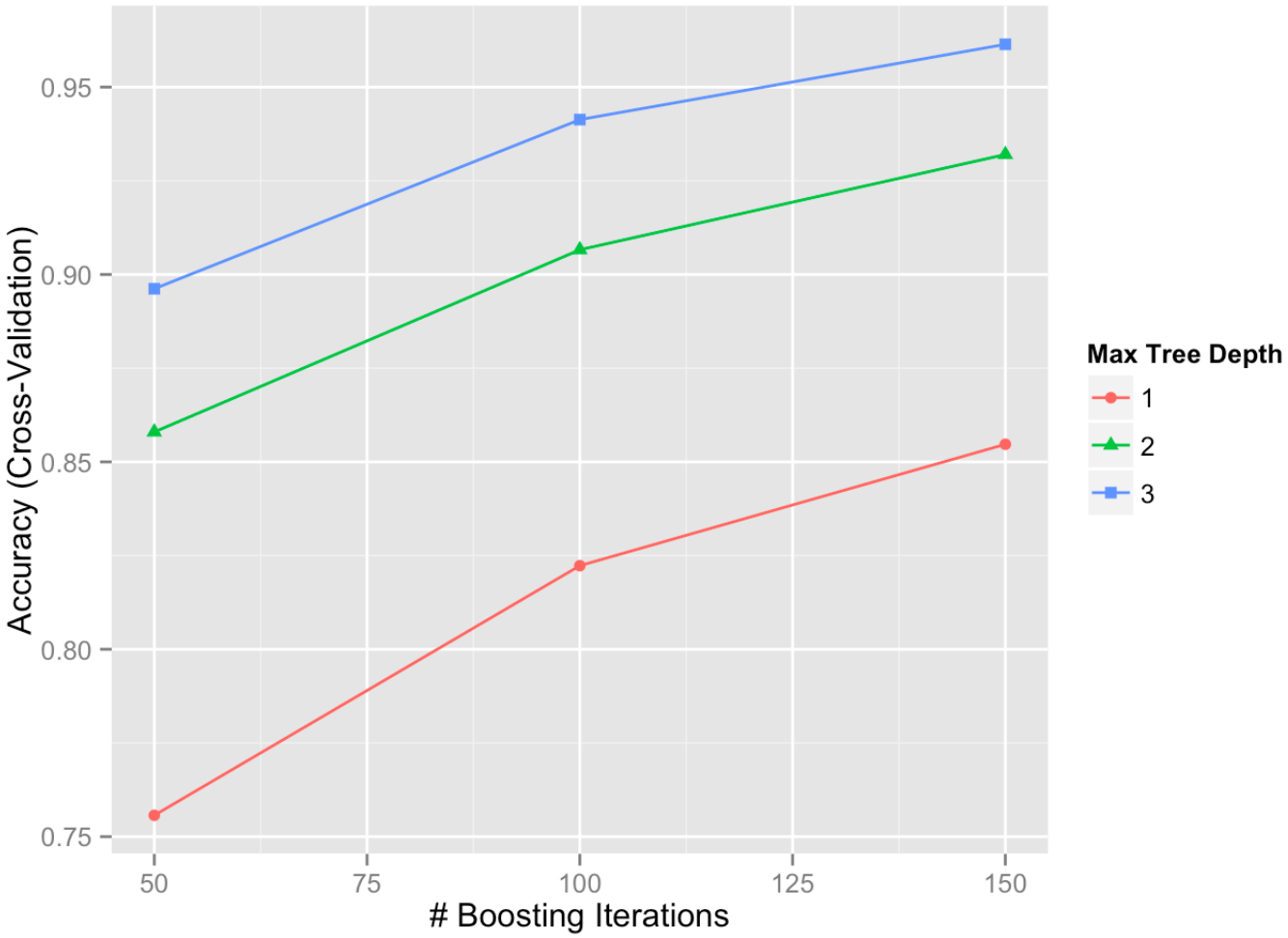
```
mypred <- predict(gbmFit1, newdata = mytest)
table(mypred,mytest$classe)
```

```
##
## mypred    A    B    C    D    E
##      A 1654   46    0    0    4
##      B   16 1048   32    7   12
##      C    2   41  978   38   12
##      D    1    1   16  913   13
##      E    1    3    0    6 1041
```

```
mypct <- sum(mypred == mytest$classe)/length(mypred)
mypct
```

```
## [1] 0.9573492
```

```
ggplot(gbmFit1)
```




# Final Model

Several additional models were fitted with increases to the number of iterations (n.trees). These showed increasng accuracy on both the training and the test subset.

For the final model, the tree complexity was tweaked, resulting in a model that produced accuracy rates of 0.9932294 and 0.9950722 on the training and test datasets, respectively. This corresponds to error rates of 0.0067706 and 0.0049278. These low error rates were corroborated by correctly predicting all 20 values in the project test dataset.

```
fitControl <- trainControl(
        method = "cv",
        number = 10)

gbmGrid <-  expand.grid(interaction.depth = c(3, 6, 9),
                        n.trees = (6:10)*25,
                        shrinkage = 0.1,
                        n.minobsinnode = 10)
set.seed(myseed)
cl <- makeCluster(no_cores)
registerDoParallel(cl)

gbmFit6 <- train(classe ~ ., data = mytrain,
                 method = "gbm",
                 preProc = c('center', 'scale'),
                 trControl = fitControl,
                 verbose = FALSE,
                 tuneGrid = gbmGrid)

stopCluster(cl)
gbmFit6
```

```
## Stochastic Gradient Boosting
##
## 13737 samples
##    51 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## Pre-processing: centered (51), scaled (51)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 12361, 12364, 12363, 12364, 12364, 12365, ...
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy   Kappa      Accuracy SD
##   3                  150      0.9616349  0.9514588  0.006842248
##   3                  175      0.9652756  0.9560671  0.004650806
##   3                  200      0.9695695  0.9614975  0.006158218
##   3                  225      0.9740822  0.9672089  0.006894092
##   3                  250      0.9776491  0.9717239  0.006469669
##   6                  150      0.9847117  0.9806604  0.003812929
##   6                  175      0.9865317  0.9829628  0.002263391
##   6                  200      0.9886427  0.9856332  0.003078720
##   6                  225      0.9901716  0.9875673  0.002622893
##   6                  250      0.9906810  0.9882119  0.003011905
##   9                  150      0.9898069  0.9871062  0.003242556
##   9                  175      0.9911173  0.9887639  0.002659054
##   9                  200      0.9921369  0.9900536  0.002657824
##   9                  225      0.9929373  0.9910662  0.002728940
##   9                  250      0.9936656  0.9919874  0.002706674
##   Kappa SD
##   0.008665074
##   0.005890586
##   0.007799035
##   0.008725071
##   0.008190090
##   0.004824927
##   0.002864236
##   0.003895456
##   0.003318553
##   0.003810396
##   0.004101952
##   0.003363741
##   0.003362146
##   0.003452209
##   0.003424120
##
## 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 = 250,
##  interaction.depth = 9, shrinkage = 0.1 and n.minobsinnode = 10.
```

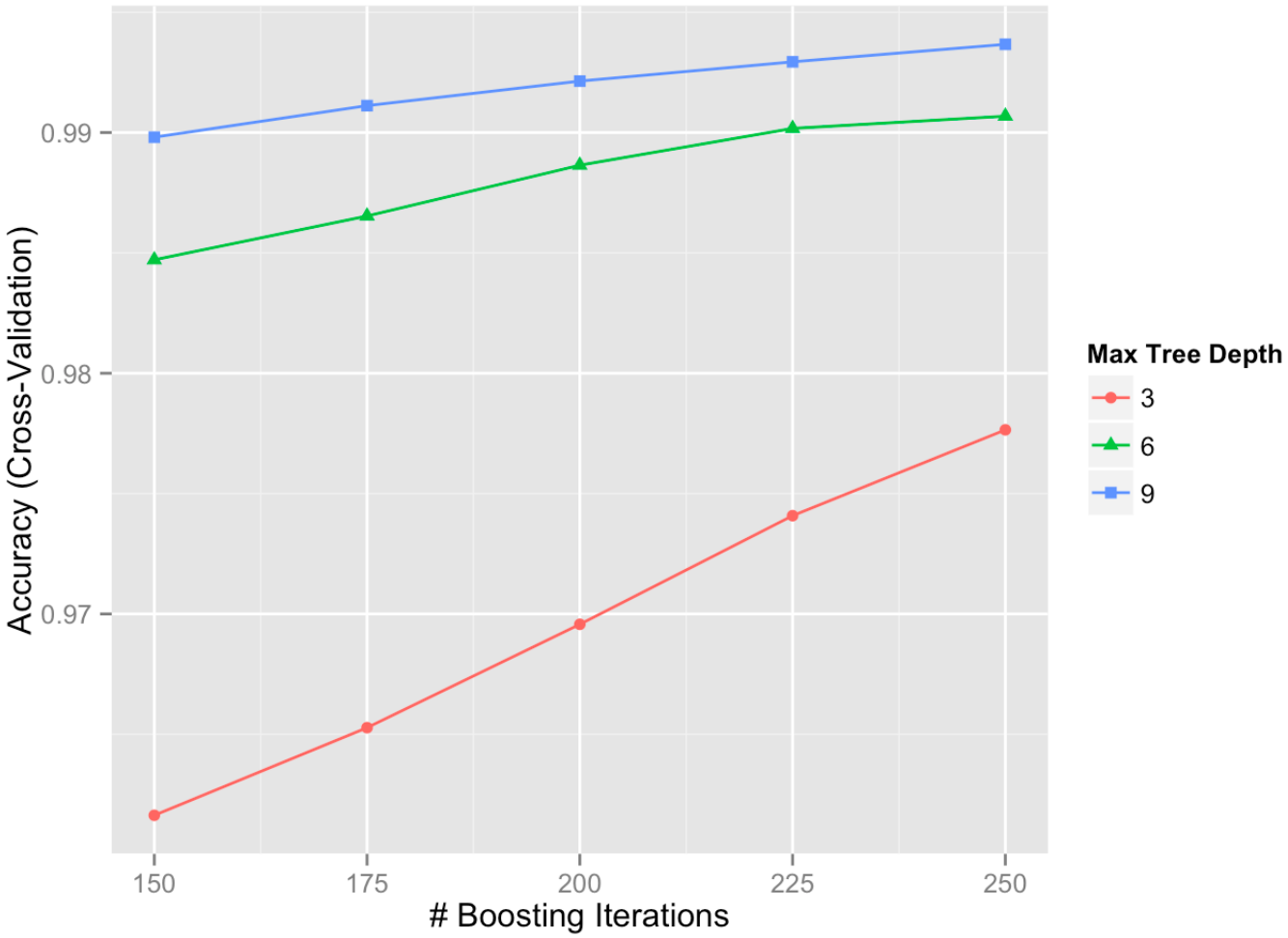
```
mypred <- predict(gbmFit6, newdata = mytest)
table(mypred,mytest$classe)
```

```
##
## mypred    A    B    C    D    E
##      A 1673    4    0    0    0
##      B    1 1132    0    0    0
##      C    0    3 1020    7    3
##      D    0    0    6  955    2
##      E    0    0    0    2 1077
```

```
mypct <- sum(mypred == mytest$classe)/length(mypred)
mypct
```

```
## [1] 0.9952421
```

```
ggplot(gbmFit6)
```


Accuracy (Cross-Validation)
0.99
0.98
0.97
150
175
200
225
250
# Boosting Iterations
Max Tree Depth
3
6
9