Machine Learning Course Project

Marc Reitz

November 12, 2018

Citation: Data for this project was sourced from: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har)

Objective: The goal of your project is to predict the manner in which they did the exercise. This is the “classe” variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Submission: Apply your machine learning algorithm to the 20 test cases available in the test data above and submit your predictions in appropriate format to the Course Project Prediction Quiz for automated grading.

Summary: For this analysis, the base data was subsetted to remove a large number of factors that predominantly recorded a value of 0 or NA. With what remained, a validation data set was carved out and three models were applied: a random forest model, a generalized boosted regression model and a combination of the two. The parallel processing methodology suggested by the instructor in the course forums was applied, which significantly reduced the time required to process the models. From these model results, the random forest model (99.24% accuracy) performed better than the GBM model (96.21%). The combo model did not appear to add any additional accuracy over the RF model, so the RF model on its own will be used for the final predictions.

Preparation

The training and test datasets are downloaded (if they have not been already) and separately read in to R. A validation subset is carved out of the training subset and used to assist with model selection.

```
if(!file.exists("./pml-training.csv"))
{
  file_url_train <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
  download.file(file_url_train, destfile = "./pml-training.csv")

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

}

library(caret, warn.conflicts = FALSE, quietly = TRUE)

training_file <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!",""))
testing_file  <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!","") )

set.seed(10101)

train_subset  <- createDataPartition(y= training_file$classe, p=.7, list = FALSE)
training      <- training_file[ train_subset,]
validation    <- training_file[-train_subset,]
testing       <- testing_file
```

The testing data set was then explored and the columns of the data set are then subsetted to remove: * Predominantly 0 or NA values. * Timestamps, user names, X * new_window, num_window columns

```
summary(training$classe)
```

```
##    A    B    C    D    E
## 3906 2658 2396 2252 2525
```

```
training    <- training  [, -which( names(training)   %in% col_name_vector )]
validation  <- validation[, -which( names(validation) %in% col_name_vector )]
testing     <- testing   [, -which( names(testing)    %in% col_name_vector )]
```

Processing the data

Parallel Processing Step 1: Configure Parallel Processing

```
library(caret, warn.conflicts = FALSE, quietly = TRUE)
library(parallel, warn.conflicts = FALSE, quietly = TRUE)
library(doParallel, warn.conflicts = FALSE, quietly = TRUE)
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
registerDoParallel(cluster)
```

Step 2: Configure trainControl Object Here, the trainControl object was configured using 5 k-folds for cross-validation.

```
fitControl <- trainControl(method = "cv",
                           number = 5,
                           allowParallel = TRUE)
```

Step 3: Develop Model

```
modFit   <- train( classe~., data = training, method = "rf", trControl = fitControl)
modFit2  <- train( classe~., data = training, method = "gbm", verbose = FALSE)

pred_1 <- predict(modFit, validation)
pred_2 <- predict(modFit2, validation)

pred_combo <- data.frame(pred_1, pred_2, classe = validation$classe)
combo_Model_fit <- train(classe~., method = "gbm", data = pred_combo, verbose = FALSE)
combPred <- predict(combo_Model_fit, pred_combo)
```

Step 4: Review results against validation set

```
# Random Forest Model
confusionMatrix(validation$classe, pred_1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    1    0    0    1
##          B    4 1135    0    0    0
##          C    0   12 1011    3    0
##          D    0    0   17  947    0
##          E    0    0    0    7 1075
##
## Overall Statistics
##
##                Accuracy : 0.9924
##                  95% CI : (0.9898, 0.9944)
##     No Information Rate : 0.2848
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9903
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9887   0.9835   0.9896   0.9991
## Specificity            0.9995   0.9992   0.9969   0.9966   0.9985
## Pos Pred Value         0.9988   0.9965   0.9854   0.9824   0.9935
## Neg Pred Value         0.9991   0.9973   0.9965   0.9980   0.9998
## Prevalence             0.2848   0.1951   0.1747   0.1626   0.1828
## Detection Rate         0.2841   0.1929   0.1718   0.1609   0.1827
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9986   0.9939   0.9902   0.9931   0.9988
```

```
# GBM Model
confusionMatrix(validation$classe, pred_2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1650   16    7    0    1
##          B   30 1072   36    1    0
##          C    0   31  977   15    3
##          D    0    6   30  916   12
##          E    1   14    3   17 1047
##
## Overall Statistics
##
##                Accuracy : 0.9621
##                  95% CI : (0.9569, 0.9668)
##     No Information Rate : 0.2856
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9521
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9816   0.9412   0.9278   0.9652   0.9849
## Specificity            0.9943   0.9859   0.9899   0.9903   0.9927
## Pos Pred Value         0.9857   0.9412   0.9522   0.9502   0.9677
## Neg Pred Value         0.9926   0.9859   0.9844   0.9933   0.9967
## Prevalence             0.2856   0.1935   0.1789   0.1613   0.1806
## Detection Rate         0.2804   0.1822   0.1660   0.1556   0.1779
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9879   0.9635   0.9588   0.9778   0.9888
```

```
# Combo Model
confusionMatrix(validation$classe, combPred)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    1    0    0    1
##          B    4 1135    0    0    0
##          C    0   12 1011    3    0
##          D    0    0   17  947    0
##          E    0    0    0    7 1075
##
## Overall Statistics
##
##                Accuracy : 0.9924
##                  95% CI : (0.9898, 0.9944)
##     No Information Rate : 0.2848
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9903
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9887   0.9835   0.9896   0.9991
## Specificity            0.9995   0.9992   0.9969   0.9966   0.9985
## Pos Pred Value         0.9988   0.9965   0.9854   0.9824   0.9935
## Neg Pred Value         0.9991   0.9973   0.9965   0.9980   0.9998
## Prevalence             0.2848   0.1951   0.1747   0.1626   0.1828
## Detection Rate         0.2841   0.1929   0.1718   0.1609   0.1827
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9986   0.9939   0.9902   0.9931   0.9988
```

Step 5: Apply model 1 to test set

```
pred_test <- predict(modFit, testing)
pred_test
```

```
##  [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
```

```
stopCluster(cluster)
registerDoSEQ()
```