Machine Learning Project - Exercise Classification

Michael R. D'Amour

Wednesday, November 19, 2014

Executive Summary

Data from the Groupware@LES organization was used to train and test a machine learning model to predict whether appropriately-monitored subjects did dumbell exercises correctly.

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

A model was built and cross validated on 1/3 of the available training set and proved to be more than 99% accurate in predicting one of the five classes of exercise styles.

Data recovery and localization
The data was downloaded from the source site, then read into memory. (NOTE: knitr had trouble handling these commands, so the data must be loaded separately and this file run in the directory that has these file local. The commands for downloading the files are shown.)

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

trainDataUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
trainDataFile <- "./pml-training.csv"
# download.file(trainDataUrl, trainDataFile)
tempPmlTrain <- read.csv(trainDataFile)

testDataUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
testDataFile <- "./pml-testing.csv"
# download.file(testDataUrl, testDataFile)
testProbSet <- read.csv(testDataFile)
```

Feature reduction and cleaning
The data set contained many features (variable columns) that were incomplete, including NAs, blanks, and #Div0s. These columns were eliminated from consideration, still leaving a large complement of appropriate data for model training.

```
# Clean up data to remove empty variables
# NOTE: the X variable was removed from the training set because of unnaturally
#   high correlation that was unrealistic.
exerFullData <- subset(tempPmlTrain, select = c("user_name",
    "new_window", "num_window", "roll_belt",
    "pitch_belt", "yaw_belt", "total_accel_belt",
    "gyros_belt_x", "gyros_belt_y", "gyros_belt_z", "accel_belt_x",
    "accel_belt_y", "accel_belt_z", "magnet_belt_x", "magnet_belt_y",
    "magnet_belt_z", "roll_arm", "pitch_arm", "yaw_arm",
    "total_accel_arm", "gyros_arm_x",
    "gyros_arm_y", "gyros_arm_z", "accel_arm_x", "accel_arm_y",
    "accel_arm_z", "magnet_arm_x", "magnet_arm_y", "magnet_arm_z",
    "roll_dumbbell",
```

```
    "pitch_dumbbell", "yaw_dumbbell", "total_accel_dumbbell",
    "gyros_dumbbell_x", "gyros_dumbbell_y", "gyros_dumbbell_z", "accel_dumbbell_x",
    "accel_dumbbell_y", "accel_dumbbell_z", "magnet_dumbbell_x", "magnet_dumbbell_y",
    "magnet_dumbbell_z", "roll_forearm", "pitch_forearm", "yaw_forearm",
    "total_accel_forearm", "gyros_forearm_x", "gyros_forearm_y",
    "gyros_forearm_z", "accel_forearm_x", "accel_forearm_y", "accel_forearm_z",
    "magnet_forearm_x", "magnet_forearm_y", "magnet_forearm_z", "classe"))
```

Subset creation for training and cross-validation testing
The data ses are loaded then partitioned using the *caret createDataPartition()* function into model training and cross-validation sets. The training data set is labeled (with the "classe" variable) to allow training of a prediction model.

testProbSet, the separate set of unlabeled/unclassified data, loaded above, was provided to do a blind test of the model. That training set was left unmodified.

```
# Partition dataset for cross validation
set.seed(050557)
exerTrainIndx <- createDataPartition(exerFullData$classe, p = .75, list = FALSE)
exerTrainBig <- select(exerFullData[exerTrainIndx, ])
exerTest <- select(exerFullData[-exerTrainIndx, ])

# Reduce training set by half
exerTrainIndx2 <- createDataPartition(exerTrainBig$classe, p = .5, list = FALSE)
exerTrain <- select(exerTrainBig[exerTrainIndx2, ])
```

This resulted in a complete set of data as follows.

Data Set Description	Set Size (obs)	(vars)	Clean Data Set Names
Total raw training set	19,622	160	tempPmlTrain
Model training subset	(38%) 7,360	56	exerTrain
Model OOB testing subset	(25%) 4,904	56	exerTest
Test problem set	20	160	testProbSet

Model choice and creation
A model was developed using the training set to predict new data (OOB data). The *randomForests* algorithm was used in the *caret train()* framework - the default *"rt"* method - based on our recent classes suggesting its superiority and the paper from UC Berkeley recommended by the instructor. It was also the method chosen by the original researchers.

```
# Model training using the "randomForest" default in train()
exerMod <- train(classe ~ .,  exerTrain)
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

Training the model, even with the reduced feature set, was found to be very compute intensive. Several smaller subsets of the training subset were tried and proved to give adequate results - high 90%s. The model given in this document is built with the *exerTrain* data which is only half of the 75% training set (i.e. 37.5% of the total available observations.)

```
# Run the prediction model on the training data to check adequacy.
# Calculate the misclassification rate using missClass modified for factors.
missClassFac = function(values,prediction){
    sum(prediction != values)/length(values)
}
trainPred <- predict(exerMod, newdata = exerTrain)
missClassFac(exerTrain$classe, trainPred)
```

```
## [1] 0
```

The misclassification proves to be adequate, allowing cross validation to proceed.

Cross Validation
Cross validation was performed on the 25% reserved testing subset, *exerTest*, to determine applicability of the model to out-of-sample data.

```
testPred <- predict(exerMod, newdata = exerTest)
missClassFac(exerTest$classe, testPred)
```

```
## [1] 0.006729
```

The expected out-of-sample error rate is below 1% (above 99% accuracy) with a suitably tight confidence interval as shown in the confusion matrix below.

```
confusionMatrix(testPred, exerTest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1391    5    0    0    0
##          B    3  938    4    0    1
##          C    0    6  849    4    0
##          D    0    0    2  799    6
##          E    1    0    0    1  894
##
## Overall Statistics
##
##                Accuracy : 0.993
##                  95% CI : (0.991, 0.995)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.991
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.997    0.988    0.993    0.994    0.992
## Specificity             0.999    0.998    0.998    0.998    1.000
## Pos Pred Value          0.996    0.992    0.988    0.990    0.998
```

```
## Neg Pred Value          0.999    0.997    0.999    0.999    0.998
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.191    0.173    0.163    0.182
## Detection Prevalence    0.285    0.193    0.175    0.165    0.183
## Balanced Accuracy       0.998    0.993    0.995    0.996    0.996
```