ML_Project

Adam

23 August 2015

Background

The goal of this project is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict the type of exercise there are performing. The data was collected performing barbell lifts correctly and incorrectly in 5 different ways.

Processing the data

The data is loaded with the the following code:

```
library(AppliedPredictiveModeling)
library(caret)
```

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

```
set.seed(1111)
pml_data <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"))
pml_test <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"))

colheadings <- c("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", "total_accel_forearm", "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_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", "roll_forearm", "pitch_forearm", "yaw_forearm", "total_accel_forearm", "classe")
colheadingstest <- c("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", "total_accel_forearm", "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_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", "roll_forearm", "pitch_forearm", "yaw_forearm", "total_accel_forearm")
pml_data_sub <- pml_data[,colheadings]
pml_test_sub <- pml_test[,colheadingstest]
pml_data_sub <- pml_data_sub[complete.cases(pml_data_sub),]
pml_data_sub$classe <- as.factor(pml_data_sub$classe)
```

Dividing the data up into a training and test set

```
#subsetting to create training and test data
pml_trainIndex = createDataPartition(pml_data_sub$classe, p = 0.7)[[1]]
pml_training = pml_data_sub[pml_trainIndex,]
pml_testing = pml_data_sub[-pml_trainIndex,]
```

Performing machine learning on the data.

Random forests techniques are used firstly to create a model based on the training data. This is then validated against the test data.

```
#Random Forest
modelRF <- train(as.factor(classe) ~ ., data = pml_training, method = "rf",metric="Accu
racy")
```

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

```
predRF <- predict(modelRF$finalModel, newdata=pml_testing)
predRF_train <- predict(modelRF$finalModel, newdata=pml_training)
confusionMatrix(predRF,pml_testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1669    7    0    0    0
##          B    4 1129    3    0    1
##          C    1    3 1021    6    3
##          D    0    0    2  958    2
##          E    0    0    0    0 1076
##
## Overall Statistics
##
##                Accuracy : 0.9946
##                  95% CI : (0.9923, 0.9963)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9931
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9912   0.9951   0.9938   0.9945
## Specificity            0.9983   0.9983   0.9973   0.9992   1.0000
## Pos Pred Value         0.9958   0.9930   0.9874   0.9958   1.0000
## Neg Pred Value         0.9988   0.9979   0.9990   0.9988   0.9988
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2836   0.1918   0.1735   0.1628   0.1828
## Detection Prevalence   0.2848   0.1932   0.1757   0.1635   0.1828
## Balanced Accuracy      0.9977   0.9948   0.9962   0.9965   0.9972
```

Based on the above the model is predicting 99.26% of the training set. If the model has not been overfitted to the training set a similar prediction rate is expected on the test set.

```
confusionMatrix(predRF_train, pml_training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3906    0    0    0    0
##          B    0 2658    0    0    0
##          C    0    0 2396    0    0
##          D    0    0    0 2252    0
##          E    0    0    0    0 2525
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

Applying the trained model to the test set the model predicts correctly 99.2% of the test set indicating that the model has not been overfitted to the test set and is an accurate predictor of the type of exercise performed.