Qualitative Activity Recognition of Weight Lifting Exercises

Carl Turner

February 26, 2016

Github Repo

A link to the Github repo for all files related to the project are found here: http://github.com/cturner3rd/QualitativeActReco (http://github.com/cturner3rd/QualitativeActReco)

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. The goal of this project was to use data from accelerometers to classify the types of movements participants made while lifting dumbells. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har)

Data

The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

Data Import and Cleansing

The testing and training motion data were read directly from the Cloudfront site. In the training set, correct movements were coded as “A” as shown in the table below (variable name “classe”.) The letters “B” through “E” were the codes for incorrect movements. The goal of the analysis was to correctly categorize the five movement types A - E in the testing data set.

Inspection of the data showed that many columns contained little useful data. Those data columns were removed prior to modeling the training data set.

```
library(lattice)
library(ggplot2)
library(caret)
library(rpart)
```

```
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
pml.training <- read.csv(url(trainUrl), stringsAsFactors = FALSE)
pml.testing <- read.csv(url(testUrl), stringsAsFactors = FALSE)
table(pml.training$classe)
```

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

```
# Data Cleaning
na_count <-sapply(pml.training, function(y) sum(length(which(is.na(y)))))
na_count <- data.frame(na_count)
table(na_count) #67 columns have 19216 NAs
```

```
## na_count
##     0 19216
##    93    67
```

```
null_count <-sapply(pml.training, function(y) sum(length(which(y==""))))
null_count <- data.frame(null_count)
table(null_count) #33 columns have 19216 ""s
```

```
## null_count
##     0 19216
##   127    33
```

```
# Select only columns that have complete data - 60 columns remaining
pml.training<-pml.training[,na_count==0 & null_count==0]
pml.testing<-pml.testing[,na_count==0 & null_count==0]
pml.training<-pml.training[,8:60] #remove first 7 cols
pml.testing<-pml.testing[,8:60] #remove first 7 cols
testing <- pml.testing
```

Training

A validation set was created from the cleansed training data set by randomly selecting 25% of the training set for use in validation. The training set was then used to fit a random forest model.

```
# Split training into training and validation 75/25
set.seed(3458)
inTrain <- createDataPartition(pml.training$classe, p = 0.75, list = FALSE)
training <- pml.training[inTrain,]
validation <- pml.training[-inTrain,]

# Random Forest: training
control <- trainControl(method = "cv", number = 5)
model_rf <- train(classe ~ ., data = training, method = "rf", trControl = control)
print(model_rf)
```

```
## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 11774, 11774, 11775, 11774, 11775
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9905560  0.9880517  0.002218701  0.002807419
##   27    0.9913713  0.9890840  0.001774226  0.002245623
##   52    0.9818592  0.9770482  0.002945547  0.003732488
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

Validation

The validation data set was used to predict the accuracy of the model on the test data set. The results are shown below. The 95% CI estimate of the accuracy of the model was 0.999% - 0.994%.

```
# Show prediction result on validation set
predict_rf_v <- predict(model_rf, validation)
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-12
```

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

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

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

```
matrix_rf_v <- confusionMatrix(validation$classe, predict_rf_v)
matrix_rf_v
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1393    1    0    0    1
##          B    3  938    7    1    0
##          C    0   10  839    6    0
##          D    0    0    7  797    0
##          E    0    1    2    1  897
##
## Overall Statistics
##
##                Accuracy : 0.9918
##                  95% CI : (0.9889, 0.9942)
##     No Information Rate : 0.2847
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9897
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9979   0.9874   0.9813   0.9901   0.9989
## Specificity            0.9994   0.9972   0.9960   0.9983   0.9990
## Pos Pred Value         0.9986   0.9884   0.9813   0.9913   0.9956
## Neg Pred Value         0.9991   0.9970   0.9960   0.9980   0.9998
## Prevalence             0.2847   0.1937   0.1743   0.1642   0.1831
## Detection Rate         0.2841   0.1913   0.1711   0.1625   0.1829
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9986   0.9923   0.9887   0.9942   0.9989
```

```
matrix_rf_v$overall[1]
```

```
##  Accuracy
## 0.9918434
```

Prediction

The model was run against the testing data set. The results are the prediction are as shown.

```
# Show prediction result on test set
predict_rf_t <- predict(model_rf, testing)
predict_rf_t
```

```
##  [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
```