Practical Machine Learning Course Project

Niisa Carter

June 2016

Practical Machine Learning Coursera Project

Prediction Assignment

Overview

One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbbell of 6 participants.

Objective

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.

One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset) Read more: http://groupware.les.inf.puc-rio.br/har#ixzz3xsbS5bVX (http://groupware.les.inf.puc-rio.br/har#ixzz3xsbS5bVX)

Datset Cleansing

Dataset Source

The data for this project come from http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har).

Load the Data

Install and load the needed R libraries:

```
library(knitr)
library (caret)
```

```
## Warning: package 'caret' was built under R version 3.2.5
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.2.5
```

```
## 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
```

Load the dataset from the given locations: 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)

```
#create the directory to store the data

if(!file.exists("./data")){dir.create("./data")}
# if the files haven't already been download, then download the training and te
sting datasets

trainingfile <- "pml-training.csv"
if (!file.exists(trainingfile)) {
    trainingdata <- read.csv(url("http://d396qusza40orc.cloudfront.net/predmach
learn/pml-training.csv"))
}
testingfile <- "pml-testing.csv"
if (!file.exists(testingfile)) {
    testingdata <- read.csv(url("http://d396qusza40orc.cloudfront.net/predmachl
earn/pml-testing.csv"))
}
```

Partition the training dataset into a training and dataset

```
subtrainingdata <- createDataPartition(trainingdata$classe, p=0.7, list = FALS
E)
subtrainingdataset <- trainingdata[subtrainingdata,]
subtestingdataset <- trainingdata[-subtrainingdata,]
```

Clean the data by removing any existing NA values as well as any near zero variances values.

```
#remove the NA values
nonas <- sapply(subtrainingdataset, function(x) mean(is.na(x))) > 0.95
subtrainingdataset <- subtrainingdataset[,nonas==FALSE]
subtestingdataset <- subtestingdataset[, nonas==FALSE]

# remove near zero variance values
nonzv <- nearZeroVar(subtrainingdataset)
subtrainingdataset <- subtrainingdataset[, -nonzv]
subtestingdataset <- subtestingdataset[, -nonzv]
```

Building the Prediction Model

Random Forest

```
# determine a model fit on the training dataset
set.seed(2016)
rftrcontrol <- trainControl(method="cv", number=3, verboseIter=FALSE)
rfmodFittrain <- train(classe ~ ., method="rf", data=subtrainingdataset, trCont
rol=rftrcontrol)
rfmodFittrain$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 41
##
##         OOB estimate of  error rate: 0.01%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3906    0    0    0    0 0.0000000000
## B    1 2657    0    0    0 0.0003762227
## C    0    0 2396    0    0 0.0000000000
## D    0    0    0 2252    0 0.0000000000
## E    0    0    0    1 2524 0.0003960396
```

```
# predict the measure of accuracy of the model fit on the training dataset
rfpredicttrain <- predict(rfmodFittrain, subtrainingdataset)
rfconfMatrixtrain <- confusionMatrix(rfpredicttrain, subtrainingdataset$classe)
rfconfMatrixtrain
```

```
## 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
```

The outcome above shows that the accuracy is 99% (sample error of 0.01%). Random Forest is accurate enough to be applied to the test data.

Apply the Prediction Model to the Test Data

The selected model, Random Forest is applied to predict the test dataset and give the quiz results.

```
rfpredicttest <- predict(rfmodFittrain, newdata=testingdata)
#rfconfMatrixtest <- confusionMatrix(rfpredicttest, testingdata$classe)
rfpredicttest
```

```
##  [1] A A A A A A A A A A A A A A A A A A A A
## Levels: A B C D E
```