Practical Machine Learning Course Project: Human Activity Recognition

Aditya Vikram

1/17/2018

Introduction

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, our goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. The goal of this project is to predict the manner in which they did the exercise.

Loading the data

```
#Load the necessary libraries
library(caret)
```

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

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

```
#Read the data files into R and replace empty/missing values by NA
train<- read.csv("pml-training.csv", sep=",", header=TRUE, na.strings = c("NA","",
'#DIV/0!'))
test<- read.csv("pml-testing.csv", sep=",", header=TRUE, na.strings = c("NA","",'#
DIV/0!'))
```

```
#Checking dimensions of the datasets
dim(train)
```

```
## [1] 19622   160
```

```
dim(test)
```

```
## [1]  20 160
```

Cleaning of data

We remove the variables with missing values.

```
train <- train[,(colSums(is.na(train)) == 0)]
dim(train)
```

```
## [1] 19622    60
```

```
test <- test[,(colSums(is.na(test)) == 0)]
dim(test)
```

```
## [1] 20 60
```

Now, our data sets have 60 variables.

Prepocess the data

```
numerical <- which(lapply(train, class) %in% "numeric")
preprocessModel <-preProcess(train[,numerical],method=c('knnImpute', 'center', 'sc
ale'))
pre_train <- predict(preprocessModel, train[,numerical])
pre_train$classe <- train$classe
pre_test <-predict(preprocessModel,test[,numerical])
```

Removing variables with values near zero

Removing the variables with values near zero as they don’t have any significance in predictions.

```
nzv <- nearZeroVar(pre_train,saveMetrics=TRUE)
pre_train <- pre_train[,nzv$nzv==FALSE]

nzv <- nearZeroVar(pre_test,saveMetrics=TRUE)
pre_test <- pre_test[,nzv$nzv==FALSE]
```

Validation set

We want a 75% observation training dataset to train our model. We will then validate it on the last 70%.

```
set.seed(100)
sample <- createDataPartition(pre_train$classe, p=0.75, list=FALSE)
training <- pre_train[sample, ]
validation <- pre_train[-sample, ]
dim(training)
```

```
## [1] 14718    28
```

```
dim(validation)
```

```
## [1] 4904   28
```

Training the model

We train a model using random forest with a cross validation of 5 folds to avoid overfitting.

```
library(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
```

```
set.seed(100)
rfmodel <- randomForest(classe ~., method="rf", data=training, trControl=trainCont
rol(method="cv", number=5))
rfmodel
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training, method = "rf",      trCont
rol = trainControl(method = "cv", number = 5))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 0.6%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4179    5    0    0    1 0.001433692
## B   10 2829    9    0    0 0.006671348
## C    0   14 2537   15    1 0.011686794
## D    0    0   18 2392    2 0.008291874
## E    1    2    3    7 2693 0.004804139
```

Cross Validation and Out-of-Sample Error

Estimate

Let's apply our training model on our testing database to check its accuracy.

Accuracy and Estimated out of sample error

```
predictions <- predict(rfmodel, validation)
confusionmatrix <- confusionMatrix(validation$classe, predictions)
confusionmatrix$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1393    2    0    0    0
##          B    2  940    7    0    0
##          C    0    3  845    7    0
##          D    0    0    8  795    1
##          E    0    0    1    1  899
```

We can notice that there are very few variables out of this model.

```
accuracy <- postResample(validation$classe, predictions)
modelAccuracy <- accuracy[[1]]
modelAccuracy
```

```
## [1] 0.9934747
```

```
out_of_sample_error <- 1 - modelAccuracy
out_of_sample_error
```

```
## [1] 0.006525285
```

The estimated accuracy of the model is 99.35% and the estimated out-of-sample error based on our fitted model applied to the cross validation dataset is 0.65%.

Application of this model on the test data set

We have already cleaned the test data set.

```
finalpred <- predict(rfmodel, pre_test)
finalpred
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```

Plots and figures

Decision tree

```
library(rpart)
library(rpart.plot)
treeModel <- rpart(classe ~ ., data=train, method="class")
prp(treeModel)
```