Practical Machine Language Project

Vasudha Upadhyaya

Wednesday, February 11, 2015

Synposis

This report analyzes to quantify how well an individual will perform a particular activity. In this project, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. People 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 (see the section on the Weight Lifting Exercise Dataset). For the report to be reproducible, I have used set.seed(). For predictions I have used random forest becuase the results are mostly accurate.

Basic settings

```
echo = TRUE  # Always make code visible
options(scipen = 1)  # Turn off scientific notations for numbers

library(caret)
## Warning: package 'caret' was built under R version 3.1.2
## Loading required package: lattice
## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 3.1.2
library(kernlab)
## Warning: package 'kernlab' was built under R version 3.1.2
library(randomForest)
## Warning: package 'randomForest' was built under R version 3.1.2
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
setInternet2(TRUE)
```

Data Cleaning and Processing

The most latest data -traing and testing are downloaded and saved as pml-traing.csv and pml-testing.csvWe read the generated csv file.To clean the data, the first row index and all colomuns with NA were removed. .

```
rm(list = ls())
# Read cleaned training and testing data

train_url<-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv"
test_url<-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv"
download.file(train_url,destfile="pml-training.csv")
download.file(test_url,destfile="pml-testing.csv")
```

```
train<-read.csv("pml-training.csv",na.strings=c("NA",""))
test<-read.csv("pml-testing.csv",na.strings=c("NA",""))

training<-train[,-c(1:7)]
set.seed(333)
```

Building data sets for training

Using 70% for training and 30% for Cross Validation. None generated for testing since that set is already provided.

```
inTrain<-createDataPartition(training$classe, p=0.75, list=FALSE)
training_train<-training[inTrain,]
training_test<-training[-inTrain,]

dim(training_train)
## [1] 14718   153
dim(test)
## [1]  20 160
training_train<-as.data.frame(training_train)
clean_train<-training_train[,colSums(is.na(training_train))==0]
clean_test<-training_test[,colSums(is.na(training_test))==0]
```

Define cross-validation experiment

```
fitControl = trainControl( method = "cv", number = 4)
# Perform the cross validation
validation <- train(classe ~ ., data = clean_train, method = "rf",
  trControl = fitControl)
bestTune<-validation$bestTune$mtry
```

Data Analysis

```
validation
## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold)
##
## Summary of sample sizes: 11039, 11039, 11038, 11038
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##    2    0.9907    0.9882  0.001953     0.002472
##   27    0.9899    0.9873  0.002754     0.003486
##   52    0.9831    0.9786  0.003644     0.004608
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

Build random forest model with full training model

```
rForest = randomForest(classe ~ ., data = clean_train,
                            mtry = bestTune)
predictTraining = predict(rForest)
table(predictTraining, clean_train$classe)
##
## predictTraining    A    B    C    D    E
##               A 4182   14    0    0    0
##               B    3 2826   17    0    0
##               C    0    8 2548   41    0
##               D    0    0    2 2369    3
##               E    0    0    0    2 2703
```

Predict testing data

```
predictTesting = predict(rForest, newdata = test)
predictTesting
##  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
```

Write the Prediction to files

```
# Function to write a vector to files
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_", i ,".txt")
    write.table(x[i], file = filename, quote = FALSE,
                row.names = FALSE, col.names = FALSE)
  }
}
# Call the function
pml_write_files(predictTesting)
```