Practical Machine Learning Project

Mayank Kumar

September 24, 2016

Project 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. 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 dumbell 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).

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)

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har). If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Data Analysis

Loading libraries

Load required libraries.

```
# load required libraries
library(caret)
```

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

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

```
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
```

```
library(rpart)
```

Data Loading

If required, download and read the training and testing sets. At time of reading, transform missing or NA data as “NA”.

```
# read training and test sets
TrainURL<-"http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
TestURL<-"http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

TrainCSV<-"pml-training.csv"
TestCSV<-"pml-testing.csv"

if(!file.exists(TrainCSV))
        download.file(TrainURL,TrainCSV)
training<-read.csv(TrainCSV,na.strings=c("NA","#DIV/0!",""))

if(!file.exists(TestCSV))
        download.file(TestURL,TestCSV)
testing<-read.csv(TestCSV,na.strings=c("NA","#DIV/0!",""))
```

Cleaning and Pre-Processing Training Set

Compare number of columns in training and testing sets

```
# compare dimensions
dim(training)
```

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

```
dim(testing)
```

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

Compare column names of training and testing sets

```
# compare column names
all.equal(sort(names(testing)[1:length(colnames(testing))-1]),sort(names(training)[1:
length(colnames(training))-1]),check.names=TRUE)
```

```
## [1] TRUE
```

Validate ‘classe’ levels

```
# check categories of classes
levels(training$classe)
```

```
## [1] "A" "B" "C" "D" "E"
```

Drop columns with less data

```
# drop columns that contain user data or meta data, not sensor data so that these col
umns don't interfere when fitting the model
training<-training[,-c(1:7)]

# drop columns with hardly any data - where less than 60% rows have data
remCol <- sapply(colnames(training), function(x) if(sum(is.na(training[,x])) > 0.60*n
row(training)) {
        return(TRUE)
}
else {
        return(FALSE)
})

training<-training[,!remCol]
dim(training)
```

```
## [1] 19622    53
```

Drop columns with near zero values

```
# drop columns with near zero values
nz<-nearZeroVar(training,saveMetrics=TRUE)
training<-training[,!nz$nzv]
```

Remove deleted columns from testing set (as well)

```
# get the columns names present in training data (except 'classe'), and extract only
those variables in testing set
testColNames<-colnames(training[,-53])
testing<-testing[,testColNames]
dim(testing)
```

```
## [1] 20 52
```

Training and Prediction

Split training data in training and validation sets

```
# set seed for reproducibility
set.seed(4196)
# split into training and test data - 70%, 30%
inTrain<-createDataPartition(y=training$classe,p=0.7,list=FALSE)
trainingTrain<-training[inTrain,]
trainingTest<-training[-inTrain,]

dim(trainingTrain); dim(trainingTest)
```

```
## [1] 13737    53
```

```
## [1] 5885   53
```

Fitting Random Forest model

Train model using training set

```
set.seed(4196)
modFitRF<-randomForest(classe~.,data=trainingTrain,ntree=1000,do.trace=FALSE)
```

Prediction with Random Forest model on validation set

```
predRF<-predict(modFitRF,trainingTest)
cmRF<-confusionMatrix(predRF,trainingTest$classe)
```

Fitting Decision Tree model

Train model using training set

```
set.seed(4196)
modFitDT<-train(classe~.,data=trainingTrain,method="rpart")
```

Prediction with Decision Tree model on validation set

```
predDT<-predict(modFitDT, trainingTest)
cmDT<-confusionMatrix(predDT, trainingTest$classe)
```

Choice of model for prediction

Let's compare the accuracy of applied models on validation set

```
# Accuracy of Random Forest model
cmRF$overall['Accuracy']
```

```
##  Accuracy
## 0.9947324
```

```
# Accuracy of Decision Tree model
cmDT$overall['Accuracy']
```

```
##  Accuracy
## 0.4985556
```

On basis of accuracy, we select Random Forest model for prediction on test set.

Predicting with Random Forest model

```
# apply Random Forst model on data given for prediction
finalPred<-predict(modFitRF,testing)
print(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
```

Conclusion

In above analysis, I've only compared Random Forest and Decision Tree models. Since, Random Forest has given very high accuracy, other models were not compared. Based on the results on validation set, model should have given good predictions on the test set as well.