Coursera - Practical Machine Learning Course Project

sof

Thursday, April 23, 2015

1. Introduction

Devices such as Jawbone Up, Nike FuelBand, and Fitbit collect a large amount of data about personal activity. The goal of this study is to interpret data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants as they perform different exercises at intentionally different levels of efficacy. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. The nature of their activities were categorized A through F representing the following actions:

- Class A - Lifting exactly according to the specification
- Class B - Throwing the elbows to the front
- Class C - Lifting the dumbbell only halfway
- Class D - Lowering the dumbbell only halfway
- Class E - Throwing the hips to the front

This study attempts to predict the nature of the activity classified A through F as explained above using the accelerometer data.

2. Source & Clean Data

2.1 Setting global default options for document

Define global settings and load all libraries required to proces the R-code chunks.

```
knitr::opts_chunk$set(echo=TRUE, results='asis')
rm(list=ls())

library(caret)
library(rattle)
library(rpart)
library(randomForest)
library(ggplot2)
```

2.2 Source and Load Data

Check to see if data has been previously downloaded, if not then download it. Load the raw training and testing datasets.

```
setwd("~/sof/datsci/coursework/8.MachineLearning/courseProject")

if (!file.exists("./data")){
```

```r
  dir.create("./data")
}

if(!file.exists('./data/pml-training.csv')){
      print("No Data File Present - Will download the file")
      download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv',
                  destfile = './data/pml-training.csv')
      }
      pml_training <- read.csv('./data/pml-training.csv')
      print('Data Loaded')
```

```
[1] "Data Loaded"
```

```r
if(!file.exists('./data/pml-testing.csv')){
      print("No Data File Present - Will download the file")
      download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv',
                    destfile = './data/pml-testing.csv')
      }
      pml_testing <- read.csv('./data/pml-testing.csv')
      print('Data Loaded')
```

```
[1] "Data Loaded"
```

2.3 Cleaning Data

The data contained many NAs and initial cleaning involved setting all these NA values to zero. Subsequently, variables that displayed near zero variance across the observations were removed from the working data as it was assumed that these would not likely be good predictors. Finally, predictors in columns 1-5 containing categorical data such as individuals names were removed from the working data as it was also assumed that these parameters would not affect the final model.

```r
# Set all NA values to zero
tmp1 <- pml_training
tmp1[is.na(tmp1)] <- 0

# Dump all variables with near zero variance, then dump col 1-5
nzv <- nearZeroVar(tmp1, saveMetrics=TRUE)
cleanTraining <- tmp1[,!(nzv$nzv)]
cleanTraining <- cleanTraining[,-c(1,2,3,4,5)]
rm(tmp1)

# Apply same cleaning to the pml_testing dataset
tmp2 <- pml_testing
tmp2[is.na(tmp2)] <- 0

# Dump same variables as those dumped in pml_training dataset
cleanTesting <- tmp2[,!(nzv$nzv)]
cleanTesting <- cleanTesting[,-c(1,2,3,4,5)]
rm(tmp2)
```

2.4 Processing Data

The cleaned training data was split into two subsets, a true training dataset and a validation / testing dataset for out-of-sample testing of the models developed in Section 3. 70% of the cleaned training observations were used for training and 30% of the observations for validation / testing purposes.

```
set.seed(1234) #To make analysis reproducable
inTrain <- createDataPartition(y=cleanTraining$classe, p=0.7, list=FALSE)
trainDat <- cleanTraining[inTrain,]
testDat <- cleanTraining[-inTrain,]
```

3. Machine Learning Model Development

Four different models are developed to attempt to use the data to predict the 'classe' variable. In all cases the mdodel is developed on the training subset of the 'cleaned training data' and then tested on the validation / testing subset of the same data, see section 2.3 above.

3.1 Using a Linear Model on Principal Components

Principal components were calculated for the training data, and a linear model was then applied to these pricipal components to try to predict the class variable. In order to use this approach the independent variable 'classe' was converted to a numeric values 1-5, which corresponded to the alphabetical values A-E. Predicted values were then rounded to the nearest integer in the 1-5 range. This was quite a cumbersome approach, and did not priduce a very accurate model (about 26% correct on the out-of-sample validation set).

```
preProc <- preProcess(trainDat[,-54], method='pca', pcaComp=20)
trainPC <- predict(preProc, trainDat[,-54])
fit1 <- train(as.numeric(trainDat$classe)~., data=trainPC, method='glm')

testPC <- predict(preProc, testDat[,-54])
pred1 <- round(predict(fit1, testPC))
pred1[pred1<=0] <- 1
pred1[pred1>5] <- 5

#table(pred1, as.numeric(testDat$classe))
perfMod1 <- confusionMatrix(pred1, as.numeric(testDat$classe))
perfMod1$table

##           Reference
## Prediction   1   2   3   4   5
##          1 336   1   0  12   7
##          2 743 271 357 129 153
##          3 537 805 654 596 505
##          4  58  62  15 196 364
##          5   0   0   0  31  53
```

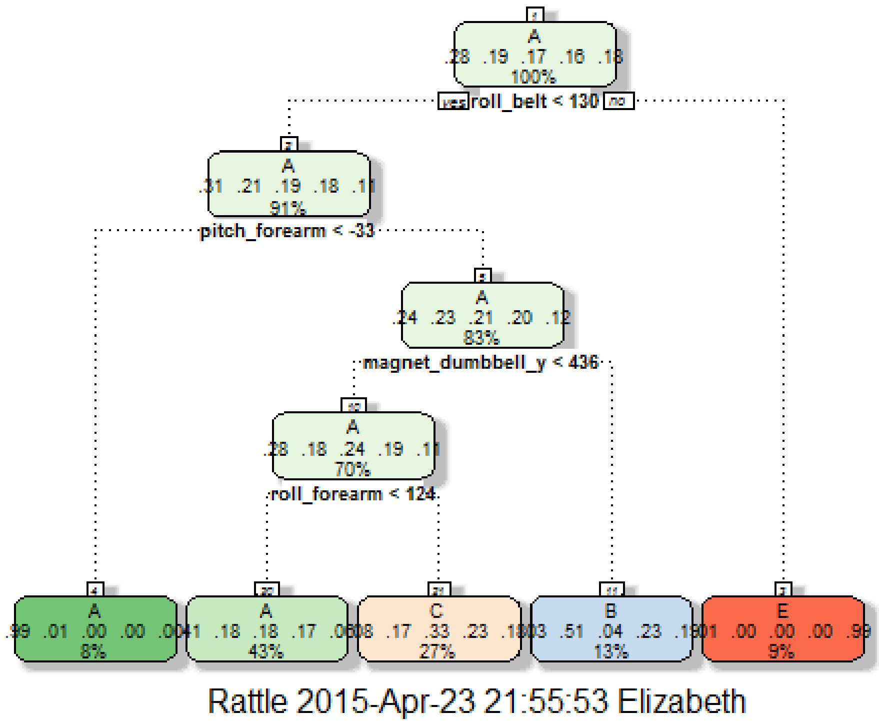
```
perfMod1$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##        0.25658        0.08751        0.24546        0.26795        0.28445
## AccuracyPValue  McnemarPValue
##        1.00000        0.00000
```

3.2 Using a Decision Tree Model

The second model used a decision tree approach to try to predict activity 'classe' from the other dependent variables in the data. The out-of-sample accuracy of Model 2 was almost two times better than the accuracy for Model 1 (about 49% correct on the out-of-sample validation set).

```
fit2 <- train(classe~., data=trainDat, method='rpart')
fancyRpartPlot(fit2$finalModel)
```



```
pred2 <- predict(fit2, testDat)
testDat$predRight <- pred2==testDat$classe; #table(pred2, testDat$classe)

#Test Model on Validation Data To Estimate Out-of-Sample Accuracy
perfMod2 <- confusionMatrix(pred2,testDat$classe)
perfMod2$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1530  486  493  452  168
##          B   35  379   31  164  145
##          C  105  274  502  348  302
##          D    0    0    0    0    0
##          E    4    0    0    0  467
```

```
perfMod2$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      4.890e-01      3.311e-01      4.762e-01      5.019e-01      2.845e-01
## AccuracyPValue  McnemarPValue
##     5.421e-240            NaN
```

3.3 Using a Random Forest with Covariate Train Control Method

A Random Forest model with 5-fold cross validation and 100 trees was attempted to try to improve the accuracy. Using this model the accuracy to predict on the the out-of-sample subset was doubled relative to the decision tree model detailed above. Using this model out-of-sample accuracy was improved to over 99% on the validation dataset.

```
fit3 <- train(classe~., data=trainDat, method='rf',
trControl=trainControl(method='cv', 5), ntree=100)
pred3 <- predict(fit3, testDat)
testDat$predRight <- pred3==testDat$classe; #table(pred3, testDat$classe)

#Test Model on Validation Data To Estimate Out-of-Sample Accuracy
perfMod3 <- confusionMatrix(pred3,testDat$classe)
perfMod3$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    2    0    0    0
##          B    0 1136    3    0    0
##          C    0    1 1023    1    0
##          D    0    0    0  963    2
##          E    0    0    0    0 1080
```

```
perfMod3$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##         0.9985         0.9981         0.9971         0.9993         0.2845
## AccuracyPValue  McnemarPValue
##         0.0000            NaN
```

3.4 Using a Random Forest with Bootstrapping Train Control Method

The final model used a Random Forest approach also with Bootstrapping as the train control method, and 100 trees. This model produced results very close to that of the previous Radom Fores model with out-of-sample prediction accuracy at over 99% accurate on the validation dataset.

```r
fit4 <- train(classe~., data=trainDat, method='rf',
trControl=trainControl(method='boot', 5), ntree=100)
pred4 <- predict(fit4, testDat)
testDat$predRight <- pred4==testDat$classe; #table(pred4, testDat$classe)

#Test Model on Validation Data To Estimate Out-of-Sample Accuracy
perfMod4 <- confusionMatrix(pred4,testDat$classe)
perfMod4$table

##           Reference
## Prediction    A    B    C    D    E
##          A 1674    3    0    0    0
##          B    0 1135    2    0    0
##          C    0    1 1024    1    0
##          D    0    0    0  963    1
##          E    0    0    0    0 1081

perfMod4$overall

##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##         0.9986         0.9983         0.9973         0.9994         0.2845
## AccuracyPValue  McnemarPValue
##         0.0000            NaN
```

4 Conculsion

Personal activity data from motion monitoring devices was analyzed to try to build a predictive model that would allow this data to be use to predict the efficacy at which that activity was being performed. Four models as detailed in Section 3 were investigated, and it was found that Random Forest models were most successful in this short study at making accurate out-of-sample predictions.

Model	Out-of-Sample Accuracy
1	0.2566
2	0.489
3	0.9985
4	0.9986

Model 3 and Model 4 built using Random Forests using some different parameters delivered the best out-of-sample accuracy on the validation subset of the data. Both models produced results that were quite comparable. Model 4 was used on the test set in the Appendix to produce submission files for Coursera for the second part of this assignment.

5. Appendix: Prepare Output Files for Submission

```r
pred5 <- predict(fit4, cleanTesting)

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)
  }
}

if (!file.exists("./testDatResults")){
  dir.create("./testDatResults")
}
setwd("~/sof/datsci/coursework/8.MachineLearning/courseProject/testDatResults
")

pml_write_files(as.character(pred5))
```