Practical Machine learning write up

Roshni Sharma

July 12, 2018

This is a write up for Prediction Assignment from Coursera for Practical Machine learning course.Background, data and Goal section is directly taken from the question.

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://web.archive.org/web/20161224072740/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

The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Goal

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Model creation and selection

Understanding data from stated site:

Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes.

Model selection approach:

I am going to start with decision tree and then random forest. If both these do not give good accuracy then i will try model based prediction or by combining predictors.

Packages, Libraries, Reproducibility

Installing packages, loading libraries, and setting the seed for reproduceability

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.4
```

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

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

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.4
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.4.4
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.4.4
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
##
## Attaching package: 'rattle'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```
set.seed(12345)
```

Load data

```
trainingdata <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"), na
testdata <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"), na.stri
```

Create training, test and validation sets

```
inTrain <- createDataPartition(y=trainingdata$classe, p=0.6, list=FALSE)
myTraining <- trainingdata[inTrain, ]
myTesting <- trainingdata[-inTrain, ]
dim(myTraining)
```

```
## [1] 11776   160
```

```
#Training set has 13737 rows and 160 features.
```

Prepocessing

Subsetting traning data for cross validation

```
train_set <- createDataPartition(y= myTraining$classe, p = 0.7, list = FALSE)
Training <- myTraining[train_set, ]
Validation <- myTraining[-train_set, ]
dim(Training); dim(Validation)
```

```
## [1] 8245  160
```

```
## [1] 3531  160
```

Feature selecting

Removing features with not much variability or impact on class, or missing values, or descriptive values which cannot contribute to the model

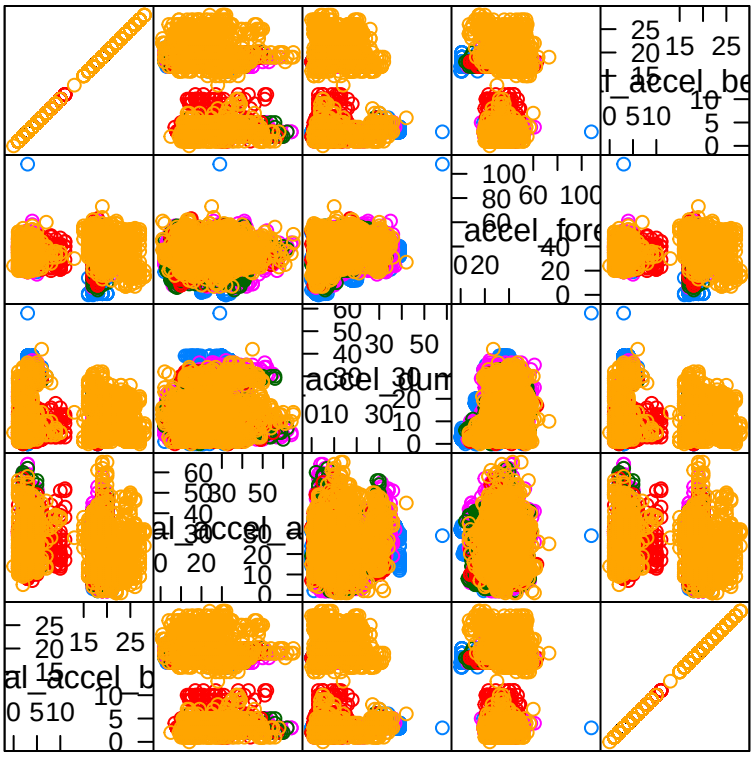
```
nzvnames<- nearZeroVar(Training)
Training <- Training[, -nzvnames]
countna <- sapply(Training, function(x) {
  sum(!(is.na(x) | x == ""))
  })
 nullcol <- names(countna[countna < 0.8 * length(Training$classe)])
 nocontricol <- c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_part_2",
    "cvtd_timestamp", "new_window", "num_window")
 excludecolumns <- c(nullcol,nocontricol)
 Training <- Training[, !names(Training) %in% excludecolumns]
 dim(Training)
```

```
## [1] 8245   53
```

```
 #doing same transformation to test and Validation data
 nzvnames<- nearZeroVar(Training)
 colnametraining <- colnames(Training)
Validation <- Validation[, colnametraining]
myTesting <- myTesting[, colnametraining]
```

Ploting predictors starting with total

```
featurePlot(x=Training[,c("total_accel_belt" ,"total_accel_arm", "total_accel_dumbbell","total_accel_fo
```

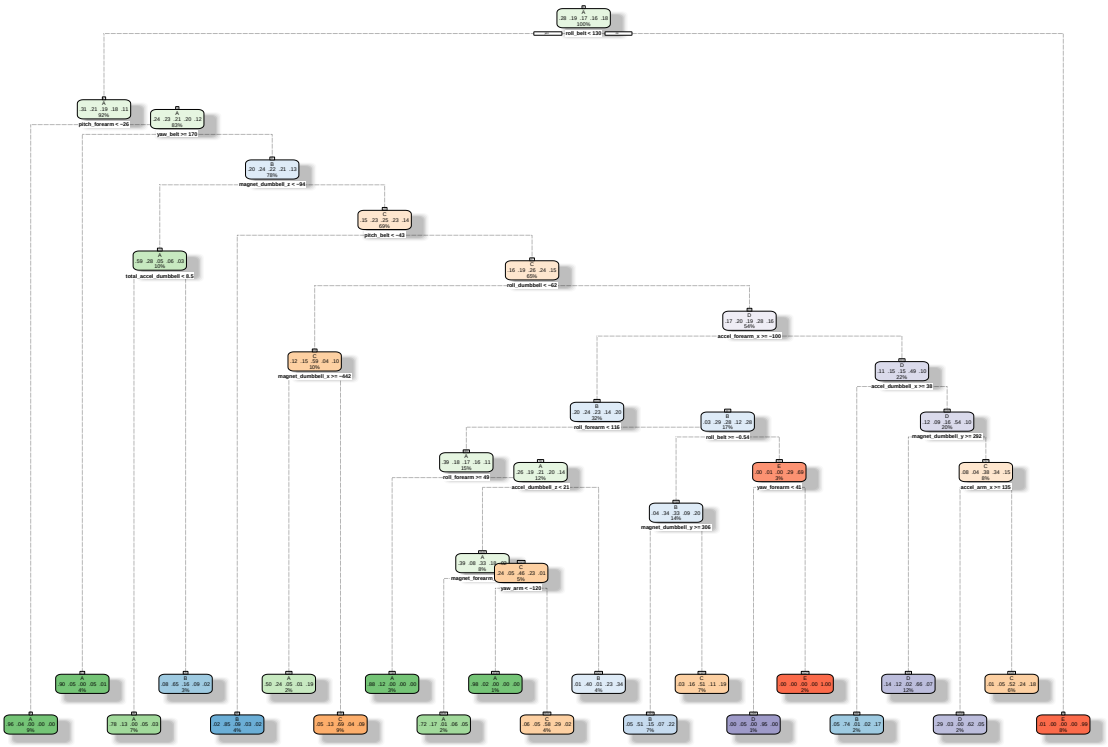


Scatter Plot Matrix

Creating Machine learning predictions - Decision Tree model1

```
model1 <- rpart(classe ~ ., data=Training, method="class")
fancyRpartPlot(model1)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



Rattle 2018−Jul−18 00:58:07 lenovo

###Predict

```
predmodel1in <-predict(model1, Training, type = "class")
predmodel1 <- predict(model1, Validation, type = "class")
```

Accuracy of MOdel1

```
# insample accuracy
confusionMatrix(predmodel1in, Training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2004  223   10   56   57
##          B   67 1017  162  156  297
##          C   78  229 1246  310  258
##          D  189  127   20  829   76
##          E    6    0    0    0  828
##
## Overall Statistics
##
##                Accuracy : 0.7185
##                  95% CI : (0.7087, 0.7282)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
```

```
##
##                   Kappa : 0.6441
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8549   0.6372   0.8665   0.6136   0.5462
## Specificity            0.9414   0.8974   0.8715   0.9402   0.9991
## Pos Pred Value         0.8528   0.5986   0.5875   0.6680   0.9928
## Neg Pred Value         0.9423   0.9115   0.9686   0.9255   0.9072
## Prevalence             0.2843   0.1936   0.1744   0.1639   0.1839
## Detection Rate         0.2431   0.1233   0.1511   0.1005   0.1004
## Detection Prevalence   0.2850   0.2061   0.2572   0.1505   0.1012
## Balanced Accuracy      0.8982   0.7673   0.8690   0.7769   0.7726
```

```
# out sample accuracy
confusionMatrix(predmodel1, Validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 865 117   9  25  23
##          B  30 383  88  56 121
##          C  31 105 513 154 124
##          D  78  78   6 344  31
##          E   0   0   0   0 350
##
## Overall Statistics
##
##                Accuracy : 0.6953
##                  95% CI : (0.6798, 0.7104)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6144
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8616   0.5608   0.8328  0.59413  0.53929
## Specificity            0.9311   0.8964   0.8580  0.93462  1.00000
## Pos Pred Value         0.8325   0.5649   0.5534  0.64060  1.00000
## Neg Pred Value         0.9442   0.8948   0.9604  0.92151  0.90600
## Prevalence             0.2843   0.1934   0.1745  0.16398  0.18380
## Detection Rate         0.2450   0.1085   0.1453  0.09742  0.09912
## Detection Prevalence   0.2943   0.1920   0.2625  0.15208  0.09912
## Balanced Accuracy      0.8963   0.7286   0.8454  0.76437  0.76965
```

Creating Machine learning predictions - RandomForest model2

```
model2 <- randomForest(classe ~. , data=Training)
```

Predict

```
predmodel2in <-predict(model2, Training, type = "class")
predmodel2 <- predict(model2, Validation, type = "class")
```

Accuracy of MOdel2

```
# insample accuracy
confusionMatrix(predmodel2in, Training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2344    0    0    0    0
##          B    0 1596    0    0    0
##          C    0    0 1438    0    0
##          D    0    0    0 1351    0
##          E    0    0    0    0 1516
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9996, 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   0.1936   0.1744   0.1639   0.1839
## Detection Rate         0.2843   0.1936   0.1744   0.1639   0.1839
## Detection Prevalence   0.2843   0.1936   0.1744   0.1639   0.1839
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
# out sample accuracy
confusionMatrix(predmodel2, Validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction    A    B    C    D    E
##          A 1004   10    0    0    0
##          B    0  668    3    0    0
##          C    0    5  612   17    1
##          D    0    0    1  562    2
##          E    0    0    0    0  646
##
## Overall Statistics
##
##                Accuracy : 0.989
##                  95% CI : (0.9849, 0.9921)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.986
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9780   0.9935   0.9706   0.9954
## Specificity            0.9960   0.9989   0.9921   0.9990   1.0000
## Pos Pred Value         0.9901   0.9955   0.9638   0.9947   1.0000
## Neg Pred Value         1.0000   0.9948   0.9986   0.9943   0.9990
## Prevalence             0.2843   0.1934   0.1745   0.1640   0.1838
## Detection Rate         0.2843   0.1892   0.1733   0.1592   0.1830
## Detection Prevalence   0.2872   0.1900   0.1798   0.1600   0.1830
## Balanced Accuracy      0.9980   0.9885   0.9928   0.9848   0.9977
```

Clearly Random Forest has much better accuracy (of 100% for insample and 98.9% for outsample)

Files for submission

```
predmodel2test <- predict(model2, myTesting, type = "class")
```

Function to generate files with predictions to submit for assignment

```
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)
  }
}

pml_write_files(predmodel2test)
```