Practical Machine Learning - Project

Amit P N

16 - Jun - 2015

This document details the Project for the course Practical Machine Learning under Data Science specialization.

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.

What you should submit

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.

- 1. Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).
- 2. You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

Modeling Details

The following sections detail about the prediction modeling, cross-valiadation and testing.

Pre-requistes

The follwoing libraries are need to be loaded if installed, otherwise need to be downloaded.

- caret package
- libraries for working with Decision trees and Randomforest
- libraries for representing decision trees.

```
setwd("C:/my files/GE Files Backup/DVD1/Learning Personal Files/MOOC Learning/GitRepo/Data_PML/DataInputFiles")

library(caret)
```

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

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

```
## Rattle: A free graphical interface for data mining with R.
## Version 3.3.0 Copyright (c) 2006-2014 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

Data Preparation

Loading and Cleaning Data

```
set.seed(7315)

# Loading the training data set - replacing all missing with "NA"
trainingset <- read.csv("C:/my files/GE Files Backup/DVD1/Learning Personal Files/MOOC Learning/GitRepo/Data_PML/DataInputFiles/pml-training.csv", na.strings=c("NA","#DIV/0!", ""))

# Loading the testing data set
testingset <- read.csv("C:/my files/GE Files Backup/DVD1/Learning Personal Files/MOOC Learning/GitRepo/Data_PML/DataInputFiles/pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))

dim(trainingset)
```

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

```
dim(testingset)
```

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

Data Cleaning

```
# Delete columns with all missing values
trainingset<-trainingset[,colSums(is.na(trainingset)) == 0]
testingset <-testingset[,colSums(is.na(testingset)) == 0]

# ignoring irrelavent data
trainingset   <-trainingset[,-c(1:7)]
testingset <-testingset[,-c(1:7)]

# new datasets:
dim(trainingset)
```

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

```
dim(testingset)
```

```
## [1] 20 53
```

```
head(trainingset,2)
```

```
##   roll_belt pitch_belt yaw_belt total_accel_belt gyros_belt_x gyros_belt_y
## 1      1.41       8.07    -94.4                3         0.00            0
## 2      1.41       8.07    -94.4                3         0.02            0
##   gyros_belt_z accel_belt_x accel_belt_y accel_belt_z magnet_belt_x
## 1        -0.02          -21            4           22            -3
## 2        -0.02          -22            4           22            -7
##   magnet_belt_y magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm
## 1           599          -313     -128      22.5    -161              34
## 2           608          -311     -128      22.5    -161              34
##   gyros_arm_x gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z
## 1        0.00        0.00       -0.02        -288         109        -123
## 2        0.02       -0.02       -0.02        -290         110        -125
##   magnet_arm_x magnet_arm_y magnet_arm_z roll_dumbbell pitch_dumbbell
## 1         -368          337          516         13.05         -70.49
## 2         -369          337          513         13.13         -70.64
##   yaw_dumbbell total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y
## 1       -84.87                   37                0            -0.02
## 2       -84.71                   37                0            -0.02
##   gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z
## 1                0             -234               47             -271
## 2                0             -233               47             -269
##   magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z roll_forearm
## 1              -559               293               -65         28.4
## 2              -555               296               -64         28.3
##   pitch_forearm yaw_forearm total_accel_forearm gyros_forearm_x
## 1         -63.9        -153                  36            0.03
## 2         -63.9        -153                  36            0.02
##   gyros_forearm_y gyros_forearm_z accel_forearm_x accel_forearm_y
## 1               0           -0.02             192             203
## 2               0           -0.02             192             203
##   accel_forearm_z magnet_forearm_x magnet_forearm_y magnet_forearm_z
## 1            -215              -17              654              476
## 2            -216              -18              661              473
##   classe
## 1      A
## 2      A
```

```
head(testingset,2)
```

```
##   roll_belt pitch_belt yaw_belt total_accel_belt gyros_belt_x gyros_belt_y
## 1    123.00      27.00    -4.75               20        -0.50        -0.02
## 2      1.02       4.87   -88.90                4        -0.06        -0.02
##   gyros_belt_z accel_belt_x accel_belt_y accel_belt_z magnet_belt_x
## 1        -0.46          -38           69         -179           -13
## 2        -0.07          -13           11           39            43
##   magnet_belt_y magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm
## 1           581          -382     40.7     -27.8     178              10
## 2           636          -309      0.0       0.0       0              38
##   gyros_arm_x gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z
## 1       -1.65        0.48       -0.18          16          38          93
## 2       -1.17        0.85       -0.43        -290         215         -90
##   magnet_arm_x magnet_arm_y magnet_arm_z roll_dumbbell pitch_dumbbell
## 1         -326          385          481        -17.74          24.96
## 2         -325          447          434         54.48         -53.70
##   yaw_dumbbell total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y
## 1       126.24                    9             0.64             0.06
## 2       -75.51                   31             0.34             0.05
##   gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z
## 1            -0.61               21              -15               81
## 2            -0.71             -153              155             -205
##   magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z roll_forearm
## 1               523              -528               -56          141
## 2              -502               388               -36          109
##   pitch_forearm yaw_forearm total_accel_forearm gyros_forearm_x
## 1          49.3         156                  33            0.74
## 2         -17.6         106                  39            1.12
##   gyros_forearm_y gyros_forearm_z accel_forearm_x accel_forearm_y
## 1           -3.34           -0.59            -110             267
## 2           -2.78           -0.18             212             297
##   accel_forearm_z magnet_forearm_x magnet_forearm_y magnet_forearm_z
## 1            -149             -714              419              617
## 2            -118             -237              791              873
##   problem_id
## 1          1
## 2          2
```

sampling - Cross-validation

Splitting training data into training and Cross-validation sets using datapartition function.

- Cross-validation is performed by sampling training data set randomly without replacement into 2 subsamples:
- subTraining data (75% of the original Training data set)
- subTesting data (25%).
- Models are fitted on the subTraining data set, and tested on the subTesting data. Most accurate model is choosen, and tested on the original Testing data set.

```
samples <- createDataPartition(y=trainingset$classe, p=0.75, list=FALSE)
subTraining <- trainingset[samples, ]
subTesting <- trainingset[-samples, ]
dim(subTraining)
```

```
## [1] 14718    53
```

```
dim(subTesting)
```

```
## [1] 4904   53
```

```
head(subTraining,2)
```

```
##   roll_belt pitch_belt yaw_belt total_accel_belt gyros_belt_x gyros_belt_y
## 1      1.41       8.07    -94.4                3         0.00            0
## 2      1.41       8.07    -94.4                3         0.02            0
##   gyros_belt_z accel_belt_x accel_belt_y accel_belt_z magnet_belt_x
## 1        -0.02          -21            4           22            -3
## 2        -0.02          -22            4           22            -7
##   magnet_belt_y magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm
## 1           599          -313     -128      22.5    -161              34
## 2           608          -311     -128      22.5    -161              34
##   gyros_arm_x gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z
## 1        0.00        0.00       -0.02        -288         109        -123
## 2        0.02       -0.02       -0.02        -290         110        -125
##   magnet_arm_x magnet_arm_y magnet_arm_z roll_dumbbell pitch_dumbbell
## 1         -368          337          516         13.05         -70.49
## 2         -369          337          513         13.13         -70.64
##   yaw_dumbbell total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y
## 1       -84.87                   37                0            -0.02
## 2       -84.71                   37                0            -0.02
##   gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z
## 1                0             -234               47             -271
## 2                0             -233               47             -269
##   magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z roll_forearm
## 1              -559               293               -65         28.4
## 2              -555               296               -64         28.3
##   pitch_forearm yaw_forearm total_accel_forearm gyros_forearm_x
## 1         -63.9        -153                  36            0.03
## 2         -63.9        -153                  36            0.02
##   gyros_forearm_y gyros_forearm_z accel_forearm_x accel_forearm_y
## 1               0           -0.02             192             203
## 2               0           -0.02             192             203
##   accel_forearm_z magnet_forearm_x magnet_forearm_y magnet_forearm_z
## 1            -215              -17              654              476
## 2            -216              -18              661              473
##   classe
## 1      A
## 2      A
```

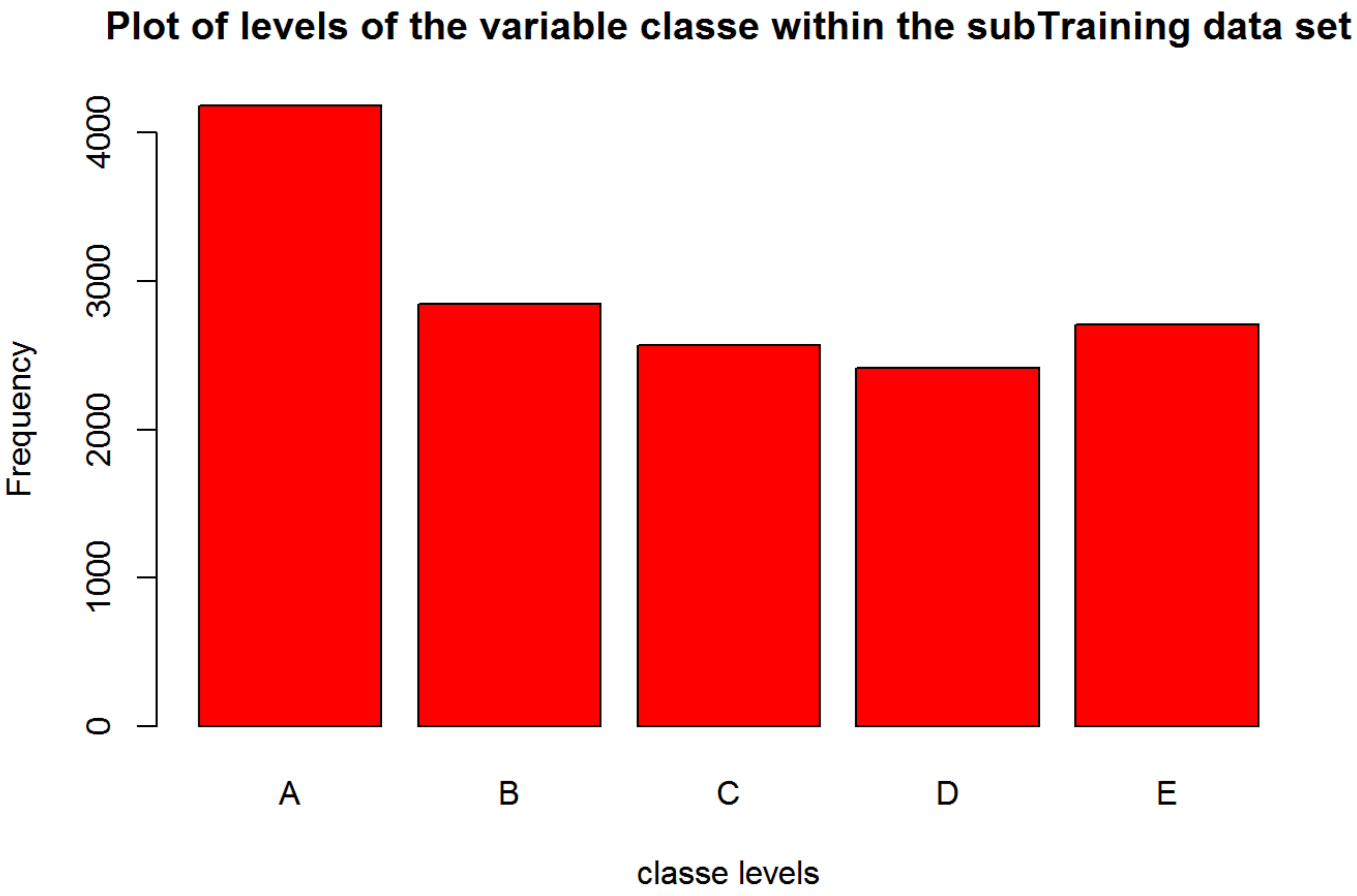
```
head(subTesting,2)
```

```
##   roll_belt pitch_belt yaw_belt total_accel_belt gyros_belt_x gyros_belt_y
## 3      1.42       8.07    -94.4                3         0.00            0
## 7      1.42       8.09    -94.4                3         0.02            0
##   gyros_belt_z accel_belt_x accel_belt_y accel_belt_z magnet_belt_x
## 3        -0.02          -20            5           23            -2
## 7        -0.02          -22            3           21            -4
##   magnet_belt_y magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm
## 3           600          -305     -128      22.5    -161              34
## 7           599          -311     -128      21.9    -161              34
##   gyros_arm_x gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z
## 3        0.02       -0.02       -0.02        -289         110        -126
## 7        0.00       -0.03        0.00        -289         111        -125
##   magnet_arm_x magnet_arm_y magnet_arm_z roll_dumbbell pitch_dumbbell
## 3         -368          344          513         12.85         -70.28
## 7         -373          336          509         13.13         -70.25
##   yaw_dumbbell total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y
## 3       -85.14                   37                0            -0.02
## 7       -85.10                   37                0            -0.02
##   gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z
## 3                0             -232               46             -270
## 7                0             -232               47             -270
##   magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z roll_forearm
## 3              -561               298               -63         28.3
## 7              -551               295               -70         27.9
##   pitch_forearm yaw_forearm total_accel_forearm gyros_forearm_x
## 3         -63.9        -152                  36            0.03
## 7         -63.9        -152                  36            0.02
##   gyros_forearm_y gyros_forearm_z accel_forearm_x accel_forearm_y
## 3           -0.02            0.00             196             204
## 7            0.00           -0.02             195             205
##   accel_forearm_z magnet_forearm_x magnet_forearm_y magnet_forearm_z
## 3            -213              -18              658              469
## 7            -215              -18              659              470
##   classe
## 3      A
## 7      A
```

Exploratory Analysis of Data

All the levels of classes with frequency

```
plot(subTraining$classe, col="red", main="Plot of levels of the variable classe within the subTrai
ning data set", xlab="classe levels", ylab="Frequency")
```



Modeling

Here two prediction (classification) models are tried

- Decision Tree modeling
- Random Forest modeling

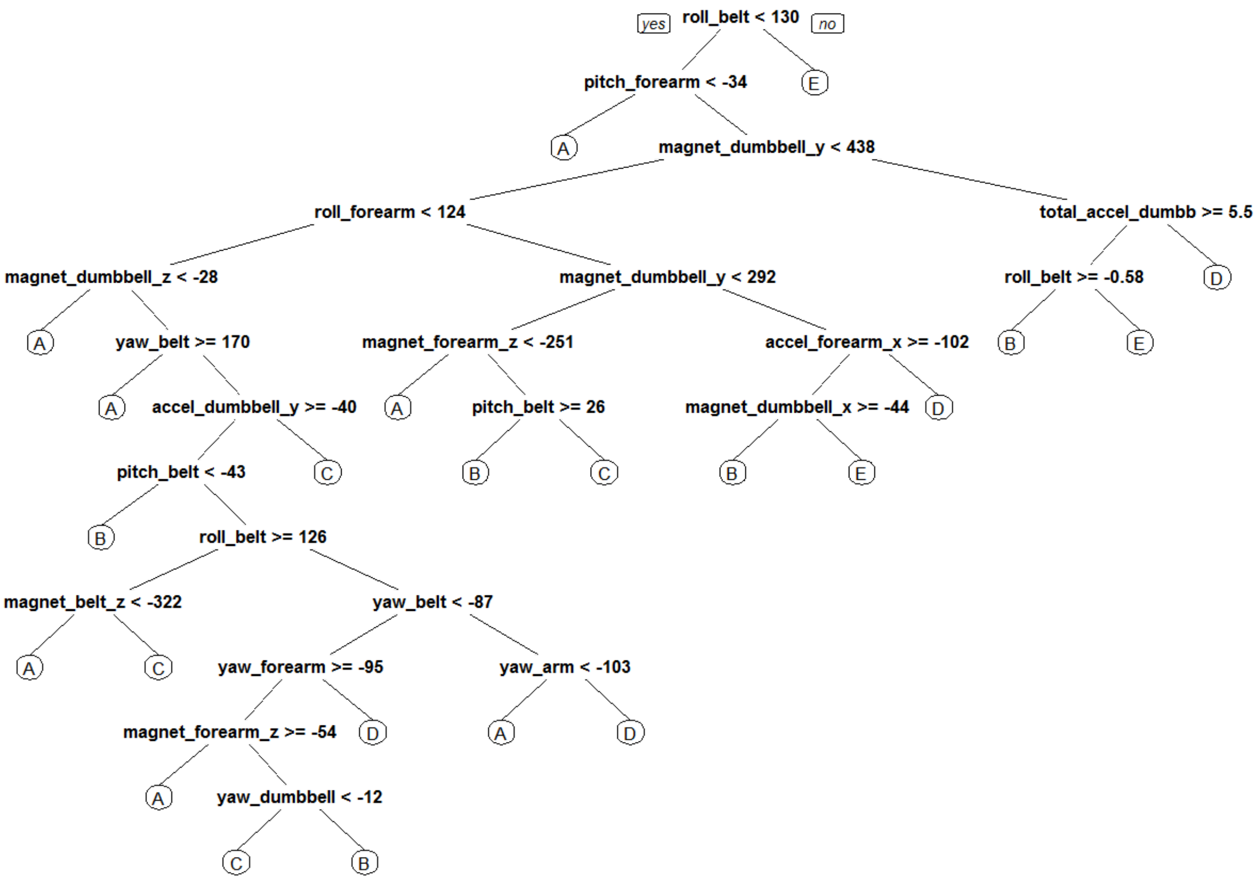
Using Decision Tree Approach

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

# Prediction for subtesting data:
prediction1 <- predict(model1, subTesting, type = "class")
```

to view the decision tree run this command

```
# Plot of the Decision Tree
rpart.plot(model1)
```



```
#fancyRpartPlot(model1)
```

prediction statistics

```
# Test results on subTesting data set:
confusionMatrix(prediction1, subTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1290  185   11   79   50
##          B   40  520   41   31   70
##          C   28   83  693  133  112
##          D   15   82   55  529   92
##          E   22   79   55   32  577
##
## Overall Statistics
##
##                Accuracy : 0.736
##                  95% CI : (0.723, 0.748)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.664
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.925    0.548    0.811    0.658    0.640
## Specificity             0.907    0.954    0.912    0.940    0.953
## Pos Pred Value          0.799    0.741    0.661    0.684    0.754
## Neg Pred Value          0.968    0.898    0.958    0.933    0.922
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.263    0.106    0.141    0.108    0.118
## Detection Prevalence    0.329    0.143    0.214    0.158    0.156
## Balanced Accuracy       0.916    0.751    0.861    0.799    0.797
```

Using Random Forest Approach

```
#Using Randome forest
model2 <- randomForest(classe ~. , data=subTraining, method="class")

# Predicting:
prediction2 <- predict(model2, subTesting, type = "class")
```

The getTree method from randomForest returns a structure. The output is shown below, with terminal nodes indicated by status code (-1).

```
# Plot of the Random Forest
rfmodeldetails <- getTree(model2, 1, labelVar=TRUE)
head(rfmodeldetails,3)
```

```
##   left daughter right daughter        split var split point status
## 1             2              3 total_accel_belt        20.5      1
## 2             4              5 accel_dumbbell_y       -38.5      1
## 3             6              7        roll_belt       129.5      1
##   prediction
## 1       <NA>
## 2       <NA>
## 3       <NA>
```

prediction statistics

```
# Test results on subTesting data set:
confusionMatrix(prediction2, subTesting$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    1    0    0    0
##          B    0  947    7    0    0
##          C    0    1  848    6    0
##          D    0    0    0  798    2
##          E    1    0    0    0  899
## 
## Overall Statistics
## 
##                Accuracy : 0.996
##                  95% CI : (0.994, 0.998)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
## 
##                   Kappa : 0.995
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.999    0.998    0.992    0.993    0.998
## Specificity             1.000    0.998    0.998    1.000    1.000
## Pos Pred Value          0.999    0.993    0.992    0.997    0.999
## Neg Pred Value          1.000    0.999    0.998    0.999    1.000
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.193    0.173    0.163    0.183
## Detection Prevalence    0.284    0.195    0.174    0.163    0.184
## Balanced Accuracy       0.999    0.998    0.995    0.996    0.999
```

Out-of-sample error

. The expected out-of-sample error - accuracy in the cross-validation data. . Accuracy is the proportion of correct classified observation . Expected accuracy is the expected accuracy in the out-of-sample data set (i.e. original testing data set).

Random Forest Approach prediction accuracy is better compared to decision tree

Prediction on TESTING Data

Prediction on testing data is done using both the modeling approaches but Random Forest Approach predictions are used for generating files.

```
# predict outcome levels on the original Testing data set using Decision Tree algorithm and Random Forest Approach
predictfinal1 <- predict(model1, testingset, type="class")
predictfinal2 <- predict(model2, testingset, type="class")
```

Writing to Files

```
# Write files for submission
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(predictfinal2)
```