Prediction Assignment Writeup

Sabin Shrestha

June 5, 2016

Project Requirement:

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.

Data: The training data for this project are available here: source: http://groupware.les.inf.puc-rio.br/har.

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Setting up the environment

```r
#install.packages("RGtk2")
#install.packages("rattle")
#install.packages('rpart')

#install.packages('lattice')
#install.packages('ggplot2')
#install.packages('rpart.plot')

library(lattice)
```

```
## Warning: package 'lattice' was built under R version 3.2.5
```

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.5
```

```r
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 3.2.5
```

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

```
## Warning: package 'rpart.plot' was built under R version 3.2.5
```

```
library(RColorBrewer)
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.2.5
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.2.5
```

```
## 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
```

Data Loading

```
pmlTraining<-read.csv("D:/Data_Science/Practical-Machine-Learning/Data/pml-training.csv", header=T, na.s
pmlTesting<-read.csv("D:/Data_Science/Practical-Machine-Learning/Data/pml-testing.csv", header=T, na.st
dim(pmlTraining)
```

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

```
dim(pmlTesting)
```

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

Data Cleansing

```
pmlTrainNoNA<-pmlTraining[, apply(pmlTraining, 2, function(x) !any(is.na(x)))]
dim(pmlTrainNoNA)
```

```
## [1] 19622    60
```

```
pmlTrainingClean<-pmlTrainNoNA[,-c(1:8)]
dim(pmlTrainingClean)
```

```
## [1] 19622    52
```

```r
pmlTrainNzv <- nearZeroVar(pmlTrainingClean, saveMetrics=TRUE)
pmlTrainingCleanNew <- pmlTrainingClean[,pmlTrainNzv$nzv==FALSE]
dim(pmlTrainingCleanNew)
```

```
## [1] 19622    52
```

```r
pmlTestingClean<-pmlTesting[,names(pmlTrainingCleanNew[,-52])]
dim(pmlTestingClean)
```

```
## [1] 20 51
```

Data Partitioning and Prediction

```r
inTrainingData<-createDataPartition(y=pmlTrainingCleanNew$classe, p=0.60,list=FALSE)
myTrainData <- pmlTrainingCleanNew[inTrainingData,]
dim(myTrainData)
```

```
## [1] 11776    52
```

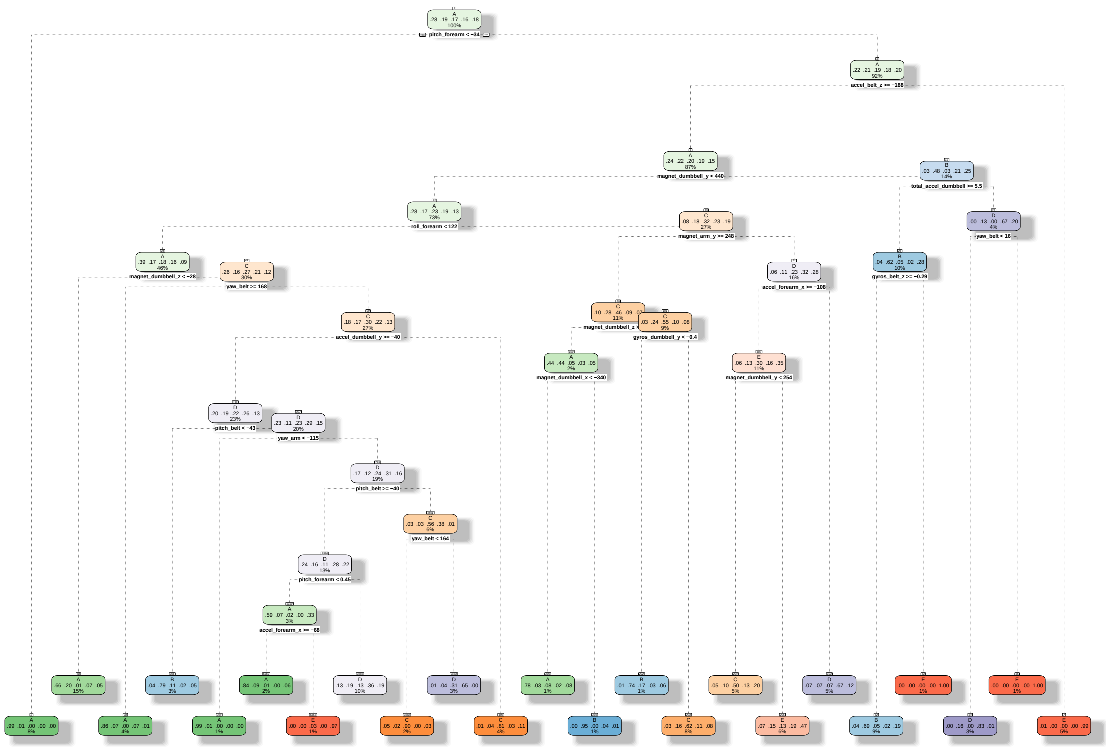
```r
myTestData <- pmlTrainingClean[-inTrainingData,]
dim(myTestData)
```

```
## [1] 7846   52
```

Results and Conclusions

```r
set.seed(12345)
modFit1 <- rpart(classe ~ ., data=myTrainData, method="class")
fancyRpartPlot(modFit1)
```

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



Rattle 2016−Jun−04 20:31:12 Sabin

```r
predict1 <- predict(modFit1, myTestData, type = "class")
confusionMatrix(predict1, myTestData$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1954  290   21  104   74
##          B   43  755   96   29  160
##          C   51  169  992  144  168
##          D  131  235  185  944  202
##          E   53   69   74   65  838
##
## Overall Statistics
##
##                Accuracy : 0.6988
##                  95% CI : (0.6885, 0.709)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6186
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8754  0.49736   0.7251   0.7341   0.5811
## Specificity            0.9129  0.94817   0.9179   0.8852   0.9592
## Pos Pred Value         0.7998  0.69714   0.6509   0.5563   0.7625
## Neg Pred Value         0.9485  0.88718   0.9405   0.9444   0.9105
## Prevalence             0.2845  0.19347   0.1744   0.1639   0.1838
## Detection Rate         0.2490  0.09623   0.1264   0.1203   0.1068
## Detection Prevalence   0.3114  0.13803   0.1942   0.2163   0.1401
## Balanced Accuracy      0.8942  0.72277   0.8215   0.8096   0.7702
```

Prediction with Regression

```r
fitControl1<-trainControl(method="cv", number=5, allowParallel=T, verbose=T)
gbmfit<-train(classe~.,data=myTrainData, method="gbm", trControl=fitControl1, verbose=F)
```

```
## Loading required package: gbm
```

```
## Warning: package 'gbm' was built under R version 3.2.5
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.1
```

```
## Loading required package: plyr
```

```
## + Fold1: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## - Fold1: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## + Fold1: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## - Fold1: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## + Fold1: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## - Fold1: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## + Fold2: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## - Fold2: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## + Fold2: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## - Fold2: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## + Fold2: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## - Fold2: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## + Fold3: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## - Fold3: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## + Fold3: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
```

```
## - Fold3: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## + Fold3: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## - Fold3: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## + Fold4: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## - Fold4: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## + Fold4: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## - Fold4: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## + Fold4: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## - Fold4: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## + Fold5: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## - Fold5: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
## + Fold5: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## - Fold5: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
## + Fold5: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## - Fold5: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
## Aggregating results
## Selecting tuning parameters
## Fitting n.trees = 150, interaction.depth = 3, shrinkage = 0.1, n.minobsinnode = 10 on full training s
```

```
gbmfit$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 51 predictors of which 42 had non-zero influence.
```

```
class(gbmfit)
```

```
## [1] "train"         "train.formula"
```

```
predict2<-predict(gbmfit, newdata=myTrainData)
confusionMatrix(predict2, myTrainData$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3320   44    0    0    1
##          B   16 2189   46    0   16
##          C   11   44 1985   65   11
##          D    0    1   19 1856   16
##          E    1    1    4    9 2121
##
## Overall Statistics
##
##                Accuracy : 0.9741
##                  95% CI : (0.9711, 0.9769)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9672
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9916   0.9605   0.9664   0.9617   0.9797
## Specificity            0.9947   0.9918   0.9865   0.9963   0.9984
## Pos Pred Value         0.9866   0.9656   0.9381   0.9810   0.9930
## Neg Pred Value         0.9967   0.9905   0.9929   0.9925   0.9954
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2819   0.1859   0.1686   0.1576   0.1801
## Detection Prevalence   0.2858   0.1925   0.1797   0.1607   0.1814
## Balanced Accuracy      0.9931   0.9761   0.9765   0.9790   0.9891
```

```r
predgbm<-predict(gbmfit, newdata=myTestData)
confusionMatrix(predgbm, myTestData$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2189   48    0    1    2
##          B   31 1419   49    4   16
##          C   10   41 1306   49   15
##          D    1    6   12 1223   17
##          E    1    4    1    9 1392
##
## Overall Statistics
##
##                Accuracy : 0.9596
##                  95% CI : (0.955, 0.9638)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9489
##  Mcnemar's Test P-Value : 4.591e-09
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9807   0.9348   0.9547   0.9510   0.9653
## Specificity            0.9909   0.9842   0.9822   0.9945   0.9977
## Pos Pred Value         0.9772   0.9342   0.9191   0.9714   0.9893
## Neg Pred Value         0.9923   0.9844   0.9904   0.9904   0.9922
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2790   0.1809   0.1665   0.1559   0.1774
## Detection Prevalence   0.2855   0.1936   0.1811   0.1605   0.1793
## Balanced Accuracy      0.9858   0.9595   0.9685   0.9728   0.9815
```

Prediction with Random Forests

```r
fitControl <- trainControl(method="cv", number=5, allowParallel=T, verbose=T)
rffit<-train(classe~.,data=myTrainData, method="rf", trControl=fitControl, verbose=F)
```

```
## + Fold1: mtry= 2
```

```
## - Fold1: mtry= 2
## + Fold1: mtry=26
## - Fold1: mtry=26
## + Fold1: mtry=51
## - Fold1: mtry=51
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry=26
## - Fold2: mtry=26
## + Fold2: mtry=51
## - Fold2: mtry=51
## + Fold3: mtry= 2
## - Fold3: mtry= 2
## + Fold3: mtry=26
## - Fold3: mtry=26
## + Fold3: mtry=51
## - Fold3: mtry=51
## + Fold4: mtry= 2
## - Fold4: mtry= 2
## + Fold4: mtry=26
## - Fold4: mtry=26
## + Fold4: mtry=51
## - Fold4: mtry=51
## + Fold5: mtry= 2
## - Fold5: mtry= 2
## + Fold5: mtry=26
## - Fold5: mtry=26
## + Fold5: mtry=51
## - Fold5: mtry=51
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 26 on full training set
```

```r
predict3<-predict(rffit, newdata=myTrainData)
confusionMatrix(predict3, myTrainData$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3348    0    0    0    0
##          B    0 2279    0    0    0
##          C    0    0 2054    0    0
##          D    0    0    0 1930    0
##          E    0    0    0    0 2165
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 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   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```r
predrf<-predict(rffit, newdata=myTestData)
confusionMatrix(predrf, myTestData$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2226   14    0    0    0
##          B    5 1492   11    2    0
##          C    0   10 1352    8    3
##          D    0    0    5 1275    6
##          E    1    2    0    1 1433
##
## Overall Statistics
##
##                Accuracy : 0.9913
##                  95% CI : (0.989, 0.9933)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.989
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9973   0.9829   0.9883   0.9914   0.9938
## Specificity            0.9975   0.9972   0.9968   0.9983   0.9994
## Pos Pred Value         0.9938   0.9881   0.9847   0.9914   0.9972
## Neg Pred Value         0.9989   0.9959   0.9975   0.9983   0.9986
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2837   0.1902   0.1723   0.1625   0.1826
## Detection Prevalence   0.2855   0.1925   0.1750   0.1639   0.1832
## Balanced Accuracy      0.9974   0.9900   0.9925   0.9949   0.9966
```

```r
predictpmlTesting<-predict(rffit, newdata=pmlTesting)
```

Output for the prediction of the 20 cases provided

```r
predictpmlTesting
```

```
##  [1] 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
```

```r
pml_write_files = function(x){
  n = length(x)
  for(i in 1:20){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
pml_write_files(predictpmlTesting)
```