Practical Machine Learning Course Project

Peer Assessments/Prediction Assignment Writeup

Executive Summary

This analysis has two key goals:

1. Predict the manner in which the sujects did the exercise referencing the "classe" variable in the training set, using cross validation, ascertaining the expected out-of-sample error, and justifying the choices made.
2. Employ the prediction model to predict 20 different test cases.

The data are sourced from: http://groupware.les.inf.puc-rio.br/har.

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

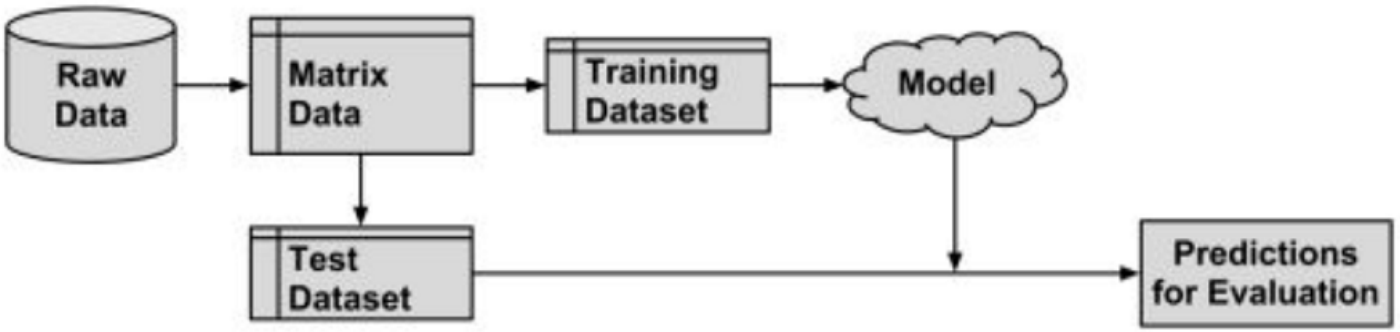
The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Cross validation is achieved by splitting the data into separate training and testing sets.

Two models are developed, based on the Random Forests and Decision Trees algorithms, the former exhibiting superior performance (with an accuracy of 0.995 and an the out-of-sample error estimated at only 0.5%). With such a high accuracy rate, it is quite likely that all of the 20 test cases will be correctly classified.

N.B. All R markdown scripts have been writtten and tested using RSudio Version 0.98.1102 and Windows 7.

A conceptual framework of the modeling logic employed is depicted in the figure below:



Preparation of the environment

1. The RStudio environment is prepared by the loading and attaching of the required packages.
2. An overall pseudo random number generator seed is set at 1234 for all code, thus ensuring reproducibility of the results obtained.
3. The training and test data sets are downloaded from their respective web addresses.
4. The excel division error strings #DIV/0! are removed and replaced with NA values. Empty strings are converted to NA values and any features that contain NA values are removed together with irrelevant variables (columns 1 to 7).

```r
if (!require("caret")) {
  install.packages("caret", repos="http://cran.rstudio.com/")
  library("caret")
}
```

```
## Loading required package: caret
```

```
## Warning: package 'caret' was built under R version 3.1.3
```

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

```r
if (!require("randomForest")) {
  install.packages("randomForest", repos="http://cran.rstudio.com/")
  library("randomForest")
}
```

```
## Loading required package: randomForest
```

```
## Warning: package 'randomForest' was built under R version 3.1.3
```

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

```r
if (!require("rpart")) {
  install.packages("rpart", repos="http://cran.rstudio.com/")
  library("rpart")
}
```

```
## Loading required package: rpart
```

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

```
## Warning: package 'rpart.plot' was built under R version 3.1.3
```

```r
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

traindata <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))
testdata <- read.csv(url(testUrl), na.strings=c("NA","#DIV/0!",""))

traindata<-traindata[,colSums(is.na(traindata)) == 0]
```

```
testdata <-testdata[,colSums(is.na(testdata)) == 0]

traindata   <-traindata[,-c(1:7)]
testdata <-testdata[,-c(1:7)]
```

Model building considerations

The key variable is “classe” which is a factor with five levels:
Class A - 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

For the derivation of this set of data, subjects were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different ways as listed above.

Class A corresponds to the correctly specified execution of the exercise, while the other classes relate to common mistakes.

Our prediction evaluations will attempt to maximize the accuracy of the model and minimize the out-of-sample error, with the inclusion of all the other variables after cleaning.

Decision-trees and random-forests algorithms will be employed to test the models. The model with the highest accuracy will be selected.

Cross-Validation by partitioning the training data set

Cross-validation will be achieved by sub-sampling the training data set randomly without replacement into 2 data sets for training (75%) and testing (25%). The models will be fitted on the training data set and tested on the testing data set. The more accurate of the two models will be selected and tested on the initial testing data set.

The cross-validation methodology is as follows:
1. Use the training set.
2. Split it into training/test sets.
3. Build a model on the training set.
4. Evaluate on the test set.
5. Repeat and average the estimated errors.

(The training data set contains 53 features and 19,622 examples. The testing data set contains 53 features and and 20 examples.)

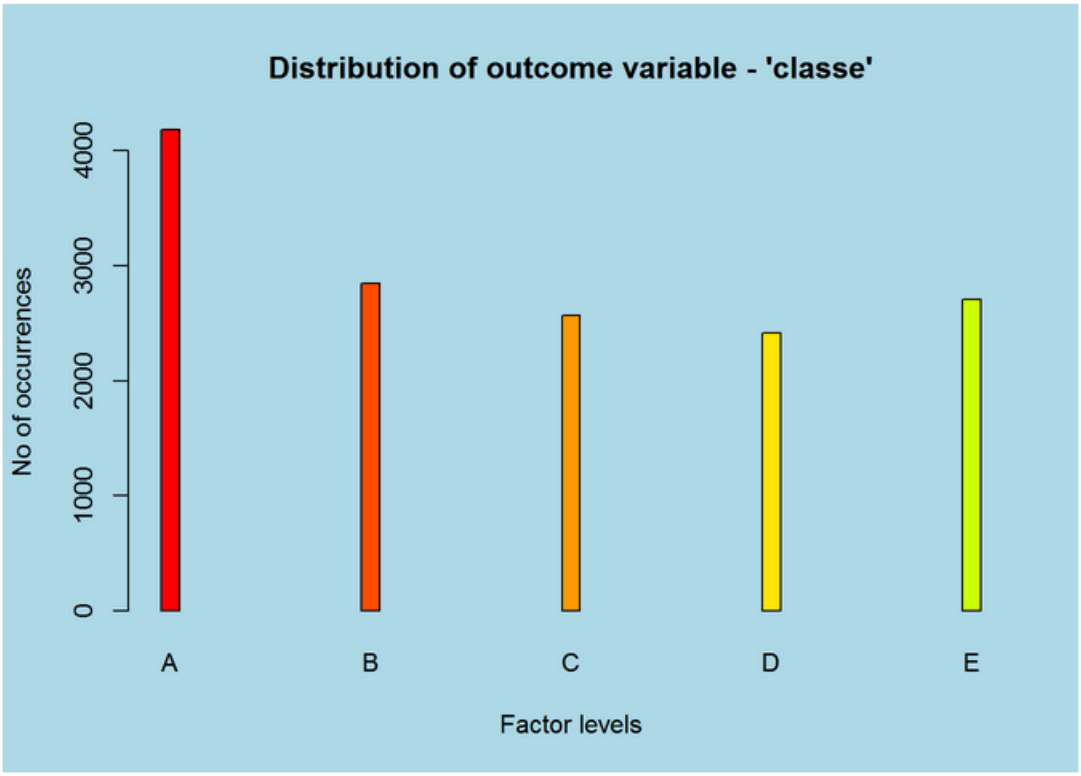
```
partsamples <- createDataPartition(y=traindata$classe, p=0.75, list=FALSE)
partTraining <- traindata[partsamples, ]
partTesting <- traindata[-partsamples, ]
```

Plotting of the outcome variable - “classe”

The variable “classe” contains 5 levels: A, B, C, D and E. A plot of the outcome variable reveals the frequency of the occurrence of the 5 levels in the training data set.

As can be seen from the plot below, the 5 levels occur with relatively similar frequencies, with Level A taking the lead at some 4,000 occurrences.

```
par(bg = "lightblue")
plot(partTraining$classe, col = rainbow(20), space=10,main="Distribution of outcome variable - 'classe'", xlab="Factor level
s", ylab="No of occurrences")
```



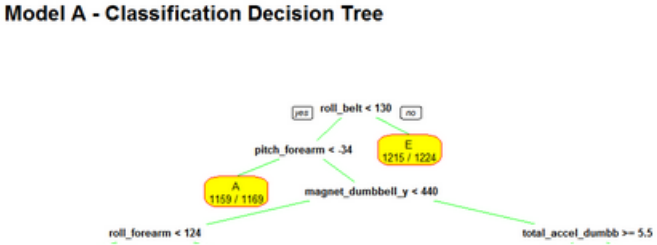
Prediction model A - decision trees

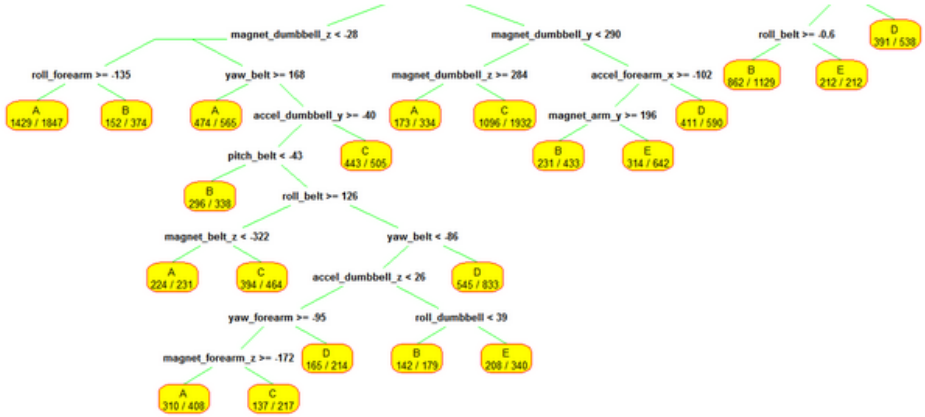
The following code creates the decision tree model along with a plot of the tree and a summary of its performance via a confusion matrix.

```
modelA <- rpart(classe ~ ., data=partTraining, method="class")

prediction1 <- predict(modelA, partTesting, type = "class")

par(bg = "white")
rpart.plot(modelA, main="Model A - Classification Decision Tree", extra=2, under=FALSE, faclen=0,
box.col="yellow", border.col="red", type=0, tweak=1,branch.lty=1,branch.col="green")
```






```
confusionMatrix(prediction1, partTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1256  155   22   42   42
##          B   34  550   64   59   76
##          C   32  145  689  111   96
##          D   54   61   52  528   45
##          E   19   38   28   64  642
##
## Overall Statistics
##
##               Accuracy : 0.7473
##                 95% CI : (0.7349, 0.7595)
##    No Information Rate : 0.2845
##    P-Value [Acc > NIR] : < 2.2e-16
##
##                  Kappa : 0.6796
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                     Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9004   0.5796   0.8058   0.6567   0.7125
## Specificity            0.9256   0.9411   0.9052   0.9483   0.9628
## Pos Pred Value         0.8279   0.7024   0.6421   0.7135   0.8116
## Neg Pred Value         0.9590   0.9032   0.9567   0.9337   0.9370
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2561   0.1122   0.1405   0.1077   0.1309
## Detection Prevalence   0.3093   0.1597   0.2188   0.1509   0.1613
## Balanced Accuracy      0.9130   0.7603   0.8555   0.8025   0.8377
```

Prediction model B - random forests

The following code creates the random forests model and a summary of its performance via a confusion matrix.

```
modelB <- randomForest(classe ~. , data=partTraining, method="class")

prediction2 <- predict(modelB, partTesting, type = "class")

confusionMatrix(prediction2, partTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    6    0    0    0
##          B    0  940    2    0    0
##          C    0    3  853    7    1
##          D    0    0    0  796    4
##          E    1    0    0    1  896
##
## Overall Statistics
##
##                Accuracy : 0.9949
##                  95% CI : (0.9925, 0.9967)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9936
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9905   0.9977   0.9900   0.9945
## Specificity            0.9983   0.9995   0.9973   0.9990   0.9995
## Pos Pred Value         0.9957   0.9979   0.9873   0.9950   0.9978
## Neg Pred Value         0.9997   0.9977   0.9995   0.9981   0.9988
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1917   0.1739   0.1623   0.1827
## Detection Prevalence   0.2855   0.1921   0.1762   0.1631   0.1831
## Balanced Accuracy      0.9988   0.9950   0.9975   0.9945   0.9970
```

Out-of-sample error

The expected out-of-sample error is given by the formula {1 - Accuracy}, hence in this model it is {1-.9947} = 0.0053, or, in other words, 0.5% is the risk of misclassification in the test data.

The kappa statistic adjusts accuracy by accounting for the possibility of a correct prediction by chance alone. Kappa values range to a maximum value of 1, which indicates perfect agreement between the model's predictions and the true values [1]. This model has a kappa of 0.9933 which again makes it unlikely that any of the 20 test candidates will be misclassified.

Reasons for selecting Model B

The Random Forest algorithm employed by Model B performed better than Decision Trees Model A.

Accuracy for Model B: 0.995 (95% CI: (0.993, 0.997))
Accuracy for Model A: 0.739 (95% CI: (0.727, 0.752))

Hence Model B is chosen and with an accuracy rate exceeding 99%; it should perform quite well on the test data set.

Assignment submission

The following result is derived on the application of the prediction model B (random forest algorithm) to the original test data set.

```
predictAnswer <- predict(modelB, testdata, type="class")
predictAnswer
```

```
##  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
```

Generation of text files with test results

```
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(predictAnswer)
```

REFERENCE

[1] Lantz, Brett (2013-10-25). Machine Learning with R (p. 303). Packt Publishing. Kindle Edition.