Practical Machine learning Course Project

Executive Summary

This report analyzes the Weight Lifting Exercises Dataset from Groupware@LES in order to predict the quality classe of the execution of various weight lifting motions based on a range of predictive variables. A CART (rpart) model was used to fit a model which can effectively predict categorical outcomes via the splitting rules of the classification tree. This was cross validated with a GBM model. Cross validation indicated that the GBM model provided more accurate results and this model was used to provide final predictions on the testing dataset.

The Data

The data was collected in an experimental study of six young health participants who were asked to complete 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).

Read more: http://groupware.les.inf.puc-rio.br/har#ixzz6SMeikHlk

Original data was seperated by a testing and training dataset. The training dataset was then divided into two subsets, training_train and training_test datasets, these subsets were used for cross validation.

Model Selection

To predict the outcome of the quality classe of the execution of the weight lifting curls, a CART (rpart) model and GBM model were tested with the outcome variable “classe” modeled as the dependent variable and 52 variables in the dataset as independent variables. Index variables and variables with many missing values were excluded. CART and GBM models are well suited for modeling categorical outcome variables and non-linear data.

The rpart and gbm function of the rpart and gbm packages, respectively, were used to fit the model to the training_train data sets. These results were then tested on the training_train data sets. The more accurate model, in this case GBM, was then selected for final predictions.

Data Analysis

Load in R Packages

```
library(caret)

## Warning: package 'caret' was built under R version 3.6.3
```

```
## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.3

library(rattle)

## Warning: package 'rattle' was built under R version 3.6.3

## Loading required package: tibble

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

library(rpart)
library(gbm)

## Loaded gbm 2.1.8
```

Loading in Data

Here data is loaded into R as data frame objects.

```
training <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!", ""))
testing <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))
```

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

Pre-processing Data

Here we clean up the data a bit, removing the first seven columns which includes the index of the dataset, "X", and other identifying variables.

```
training <- training[, -(1:7)]
testing <- testing[, -(1:7)]
```

We then check for variables with zero variance, many variables have near zero variance, though none have zero variance, so we leave them in.

```
nearZeroVar(training, saveMetrics = TRUE)
```

We also check for variables with many NAs and then remove those columns from the data set

```
sapply(training, function(x)sum(is.na(x)))
#results are hidden but several variables have over 19,000 missing values. So
we remove them here:
trainIndex <- 0
for(i in 1:ncol(training)) {
        if(sum(is.na(training[, i])) > 19000){
                trainIndex <- c(trainIndex, i)
        }
}
trainIndex <- trainIndex[-1]
training <- training[, -trainIndex]
dim(training)
# variables were removed
```

Let's now do the same preprocessing on the testing dataset.

```
cleanVar <- names(testing) %in% names(training)
testing <- testing[cleanVar]
```

Cross validation

To perform cross validation, we divide the training data set into two subsets. We then train two different models, a CART model and a GBM model on the training subset and then test these on the testing subset. The model with less error is then chosen to be used to predict classe on the testing dataset.

```
trainIndex <- createDataPartition(y=training$classe, p=0.75, list=FALSE)
training_train <- training[trainIndex, ]
training_test <- training[-trainIndex, ]
```

Fitting the Model - CART model

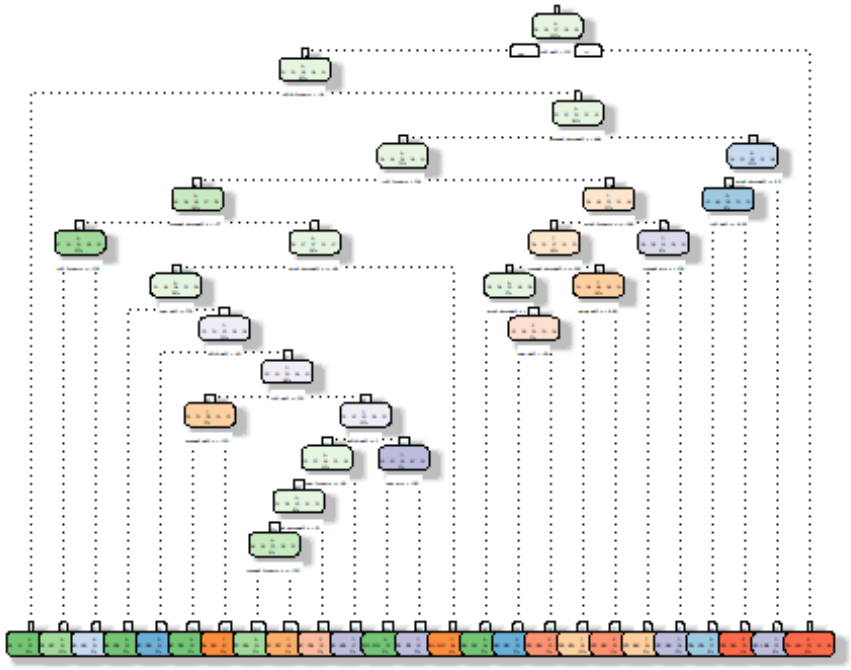
We now fit a CART (rpart) model on the training data with the rparttt function from the rpart package.

```
modFit_rpart <- rpart(classe ~ ., data = training_train, method = "class")
```

And let's plot the classification tree in a fancy rpart plot with the rattle package.

```
fancyRpartPlot(modFit_rpart)

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



Let's next predict the values of the classe variable on the training_test set and store these in the variable predictions_rpart. We then look at the accuracy of these predictions in a confusion matrix.

```
predictions_rpart <- predict(modFit_rpart, training_test, type = "class")
confusionMatrix(predictions_rpart, training_test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1293  149   18   48   10
##          B   36  490   41   48   60
##          C   30  185  738   80  110
##          D   23   60   57  563   77
##          E   13   65    1   65  644
##
## Overall Statistics
##
##                Accuracy : 0.7602
##                  95% CI : (0.748, 0.7721)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6961
##
##  Mcnemar's Test P-Value : < 2.2e-16
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9269  0.51633   0.8632   0.7002   0.7148
## Specificity            0.9359  0.95322   0.9000   0.9471   0.9640
## Pos Pred Value         0.8518  0.72593   0.6457   0.7218   0.8173
## Neg Pred Value         0.9699  0.89146   0.9689   0.9416   0.9376
## Prevalence             0.2845  0.19352   0.1743   0.1639   0.1837
## Detection Rate         0.2637  0.09992   0.1505   0.1148   0.1313
## Detection Prevalence   0.3095  0.13764   0.2331   0.1591   0.1607
## Balanced Accuracy      0.9314  0.73478   0.8816   0.8237   0.8394
```

The accuracy is estimated to be nearly 75%, indicating that the out of sample for this model is estimated to be around 25%.

Fitting the Model - GBM model

We next fit a GBM model to the training_train data with a specified multinomial distribution.

```
modFit_gbm <- gbm(classe ~ ., data = training_train, distribution =
"multinomial")

## Warning: Setting `distribution = "multinomial"` is ill-advised as it is
## currently broken. It exists only for backwards compatibility. Use at your
own
## risk.
```

We then use this model to predict the classe values for the training_test data subset and check the accuracy of these results in a confusion matrix.

```
predictions_gbm <- predict.gbm(modFit_gbm, training_test, type = "response")

## Using 100 trees...

predictions_gbm_final <-
factor(colnames(predictions_gbm)[apply(predictions_gbm, 1, which.max)])
confusionMatrix(predictions_gbm_final, training_test$classe)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1244  123   19   44   28
##          B   41  687   60    7   75
##          C   39   80  752   82   45
##          D   53   48   22  647   37
##          E   18   11    2   24  716
##
## Overall Statistics
```

```
##
##                Accuracy : 0.825
##                  95% CI : (0.8141, 0.8356)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7785
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8918   0.7239   0.8795   0.8047   0.7947
## Specificity            0.9390   0.9537   0.9392   0.9610   0.9863
## Pos Pred Value         0.8532   0.7897   0.7535   0.8017   0.9287
## Neg Pred Value         0.9562   0.9351   0.9736   0.9617   0.9552
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2537   0.1401   0.1533   0.1319   0.1460
## Detection Prevalence   0.2973   0.1774   0.2035   0.1646   0.1572
## Balanced Accuracy      0.9154   0.8388   0.9094   0.8829   0.8905
```

The accuracy is estimated to be over 80% with the gbm model, indicating that the estimated out of sample error is less than 20%. Hence, these results indicate that the gbm model is a more accurate predictor of classe values than the CART model.

Final Predictions

Here we fit the testing data to the gbm model, which was determined to be the more accurate model in the cross validation process, to get the predicted outcomes for this second dataset.

```
predictions_final <- predict.gbm(modFit_gbm, testing, type = "response")

## Using 100 trees...

predictions_final <-
factor(colnames(predictions_final)[apply(predictions_final, 1, which.max)])
print(predictions_final)

##  [1] D A A A A E D D A A B C B A E A A B A B
## Levels: A B C D E
```

Conclusion

In this report we built a prediction model to estimate the quality classe of an executed weight lifting curl by fitting a CART model and a GBM model on the Weight Lifting Exercises Dataset. We fit both models on a training subset of the original training dataset and then performed cross validation on a testing subset of the original training dataset.

Results of cross validation indicated that the GBM model provided more accurate predictions than the CART model. Finally, the GBM model was used to provide final predictions of classe values on the testing dataset.