Machine Learning Project 1

Dennis Oriaifo

July 2nd, 2017

```
## 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


## GOAL

# The goal of the project is to predict the manner in which the participants did the exercise.
# This is the "classe" variable in the training set. Other variables may be used to predict the outcome
# A report must be written, describing how the model was built, how cross validation was used,
# what the expected out of sample error is, and why specific choices were made.


## DATA

# The training data can be found here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training
# The test data can be found here: 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.
# 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.


## DATA PREPROCESSING

# Clear console
rm(list=ls())
cat('\014')
```

```
# Load packages
library(caret)
```

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

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

```
library(corrplot)
library(caTools)
library(ggplot2)
library(knitr)
library(plyr)
library(randomForest)
```

```
## 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
```

```
# Download the data
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
              destfile = "training_set.csv")
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
              destfile = "test_set.csv")

# Create datasets
training_set = read.csv("training_set.csv")
test_set = read.csv("test_set.csv")

# Remove personal ID fields - not relevant to the model
training_set = training_set[, -(1:5)]
test_set  = test_set[, -(1:5)]

# Classify all columns with blanks and errors as NA for easier removal
training_set[training_set == ""] = NA
training_set[training_set =="#DIV/0!"] = NA

test_set[test_set == ""] = NA
test_set[test_set =="#DIV/0!"] = NA

# Remove all NAs from dataset
training_setNAs = sapply(training_set, function(x) mean(is.na(x))) > 0.95
training_set = training_set[, training_setNAs==FALSE]
test_set  = test_set[, training_setNAs==FALSE]

# Split dataset for training and testing
set.seed(123)
split = sample.split(training_set$classe, SplitRatio = 0.8)
training_setSplit = subset(training_set, split == TRUE)
test_setSplit = subset(training_set, split == FALSE)

## MODEL SELECTION

# There are several regression models available to analyze our data -
# Multiple Linear Regression, Polynomial Regression, Support Vector Regression, Decision Tree
# and Random Forest However, due to the presence of a lot of noise in the data and potential
# nonlinearity I will be using a Random Forest. I believe RF will deal well with non linearity
# in the data without the need for interaction terms, data transformations (as in the case of
# Polynomial Regression) and will be more accurate than Multiple/Polynomial/Support Vector Regression.


# Create RF Model - Train model
```

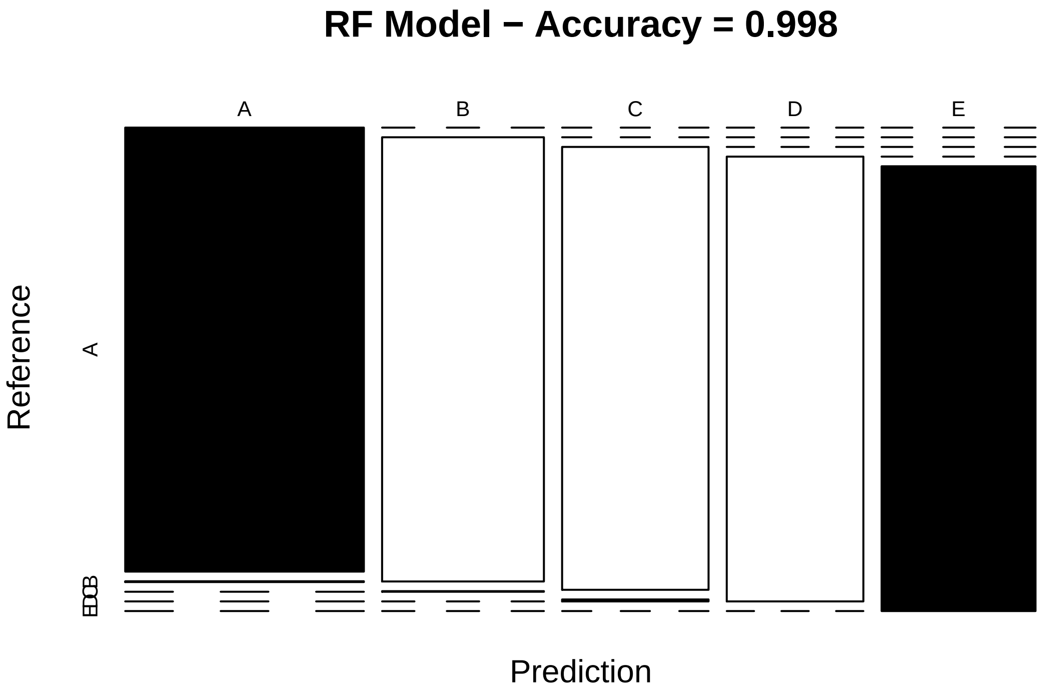
```
set.seed(123)
RFModel = randomForest(classe ~ ., data=training_setSplit)

# Applying RF to test split - Test model
RFPredictor = predict(RFModel, test_setSplit, type = "class")

# Constructing a Confusion Matrix
RFConfMatrix = confusionMatrix(RFPredictor, test_setSplit$classe)
RFConfMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    2    0    0    0
##          B    0  757    1    0    0
##          C    0    0  683    3    0
##          D    0    0    0  640    0
##          E    0    0    0    0  721
##
## Overall Statistics
##
##                Accuracy : 0.9985
##                  95% CI : (0.9967, 0.9994)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9981
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9974   0.9985   0.9953   1.0000
## Specificity            0.9993   0.9997   0.9991   1.0000   1.0000
## Pos Pred Value         0.9982   0.9987   0.9956   1.0000   1.0000
## Neg Pred Value         1.0000   0.9994   0.9997   0.9991   1.0000
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1930   0.1741   0.1631   0.1838
## Detection Prevalence   0.2850   0.1932   0.1749   0.1631   0.1838
## Balanced Accuracy      0.9996   0.9985   0.9988   0.9977   1.0000
```

```
#Out of Sample Error
1 - RFConfMatrix$overall['Accuracy']
```

```
##    Accuracy
## 0.001529442
```

```
# Plotting a Confusion Matrix
plot(RFConfMatrix$table, col = RFConfMatrix$byClass,
     main = paste("RF Model - Accuracy =",
                  round(RFConfMatrix$overall['Accuracy'], 3)))
```



```
# The Random Forest model yields 99% accuracy which is surprisingly very good.

# Applying model to Test dataset
# RFPredictorTest = predict(RFModel, test_set, type = "class")
# RFPredictorTest


#########
```