Practical Machine Learning Course Project Report

Enzo

The Current report, describes the steps involved in the creation of a learning machine to predict the activity from devices such as Jawbone Up, Nike FuelBand, and Fitbit. The Current report shows the steps required to define the model: getting the data, partition the data (in training and testing), perform the first model using a simple Decision Tree, and then a more sophisticate model the Random Forest.

1 Getting and Loading in Memory the Data Set

The data set was provided in the training.csv file. It was defined the working directory for all the file for this report. Then the data set was load into the memory, ready to be analysed.

```
setwd("C:/Users/cv/Desktop/learningmachines/")
Data<-read.csv("training.csv")
```

2 Partition the DataSet

The Data Set was divided into two different sets, one for the model training 60% and one for the testing 40%.

```
library(caret)
```

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

```
index<-createDataPartition(y=Data$classe, p=0.6, list=FALSE)
Training<-Data[index,]
Testing<-Data[-index,]
```

3 Cleaning the dataSet

The data set has 160 variables, but many of them are not relevant for the model. Analyzing the variables, have been identified the variables: with nearly no variance those full of NA for most of the observations. and the ones with no formal meaning for the model (like X, timestamp, user,. . .)
And all those variables have been removed from the training and testing sets.

Removing the no-variance variables:

```
index2<-nearZeroVar(Training)
```

Removing the variables that are no formal meaning for the model and have most of the observations with No valid values (NA):

```
index2<-c(index2,grep("kurtosis",names(Training)),grep("kewness",names(Training)),grep("X",names(Traini

New_Traing<-Training[,-index2]
New_Testing_Data<-Testing[,-index2]
```

4 The First Algorithms for Prediction: Decision Tree

The first model is the **Decision Tree**, trained using the training set,

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

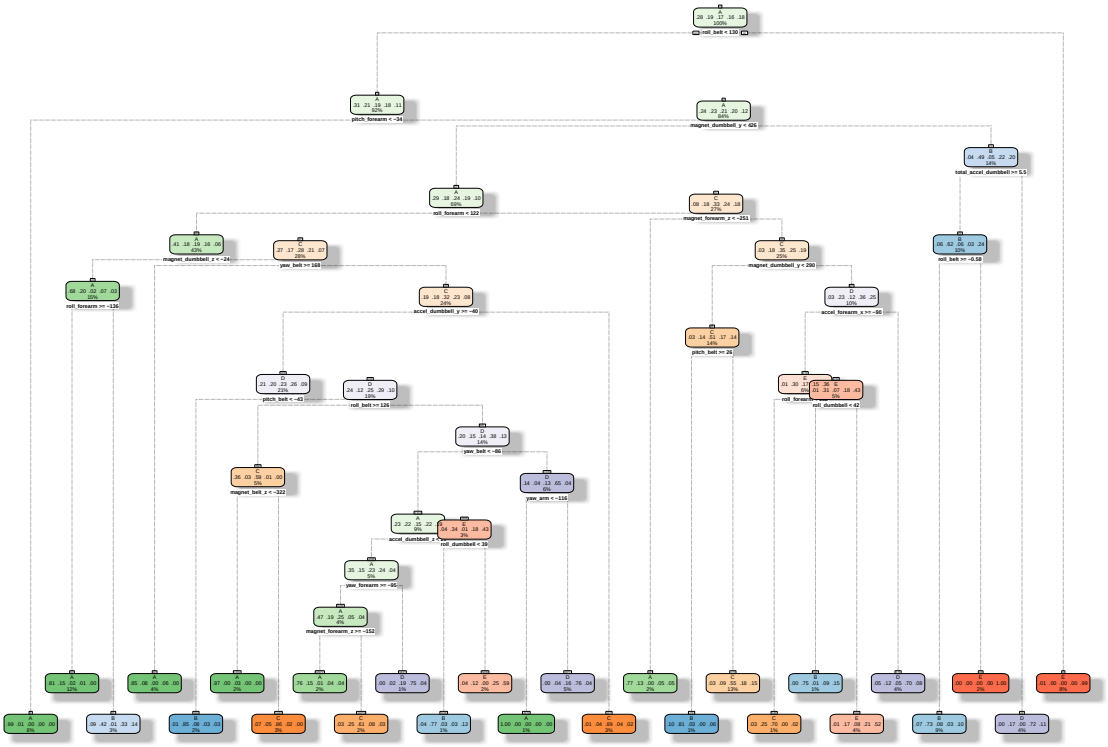
```
## Rattle: A free graphical interface for data mining with R.
## Version 3.4.1 Copyright (c) 2006-2014 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
mod1<-rpart(classe ~ ., data=New_Traing, method="class")
```

The following is the graph showing the classification tree output of the algorithm.

```
fancyRpartPlot(mod1)
```

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



Rattle 2015−Aug−19 23:08:01 cv

The following is the **Confusion Matrix** of the model. As can be seen the accuracy is far away from the 100%, and the classification is not very precise.

```
pred1<-predict(mod1,New_Testing_Data,type="class")
confusionMatrix(pred1, New_Testing_Data$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2042  249   28   50   27
##          B   90  927   66  112  122
##          C   57  164 1163  194  175
##          D   27  106   78  815   80
##          E   16   72   33  115 1038
##
## Overall Statistics
##
##                Accuracy : 0.7628
##                  95% CI : (0.7532, 0.7722)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6993
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9149   0.6107   0.8501   0.6337   0.7198
## Specificity            0.9369   0.9384   0.9089   0.9556   0.9631
## Pos Pred Value         0.8523   0.7039   0.6634   0.7369   0.8148
## Neg Pred Value         0.9651   0.9095   0.9664   0.9301   0.9385
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2603   0.1181   0.1482   0.1039   0.1323
## Detection Prevalence   0.3054   0.1679   0.2234   0.1410   0.1624
## Balanced Accuracy      0.9259   0.7745   0.8795   0.7947   0.8415
```

5 The Second Algorithms for Prediction: Random Forest

The second model is the **Random Forest**, trained as before using the training set,

```
library(randomForest)
```

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

```
mod2<-randomForest(classe ~ ., data=New_Traing, method="class")
```

The following is the **Confusion Matrix** of the model. As can be seen the accuracy is nearly to the 100%, and the classification is very precise.

```r
pred2<-predict(mod2,New_Testing_Data,type="class")
confusionMatrix(pred2, New_Testing_Data$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2230    5    0    0    0
##          B    0 1510    7    0    0
##          C    2    3 1359   15    1
##          D    0    0    2 1271   11
##          E    0    0    0    0 1430
##
## Overall Statistics
##
##                Accuracy : 0.9941
##                  95% CI : (0.9922, 0.9957)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9926
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9947   0.9934   0.9883   0.9917
## Specificity            0.9991   0.9989   0.9968   0.9980   1.0000
## Pos Pred Value         0.9978   0.9954   0.9848   0.9899   1.0000
## Neg Pred Value         0.9996   0.9987   0.9986   0.9977   0.9981
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1925   0.1732   0.1620   0.1823
## Detection Prevalence   0.2849   0.1933   0.1759   0.1637   0.1823
## Balanced Accuracy      0.9991   0.9968   0.9951   0.9932   0.9958
```

This model was able to identified all the 20 classification required for the second part of the project.