Classification of Accelerometer Data with Machine Learning

Neil Kutty

1.

Load Necessary Libraries

```r
library(caret)
library(GGally)
library(rpart)
library(rpart.plot)
library(party)
library(RGtk2)
library(rattle)
library(xgboost)
library(formattable)
library(dplyr)
library(tidyr)
library(tibble)
library(ggthemes)
```

2.

Download Data

```r
train <- read.csv('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv')
test <- read.csv('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv')
```

3.

Create training / test partition for model validation

```r
inTrain <- createDataPartition(train$classe, p=0.75, list = F)
training <- train[inTrain,]
testing <- train[-inTrain,]
```

4.

Identify and remove near zero variance predictors

```r
#Find near zero variance predictors
nzvs <- nearZeroVar(training, saveMetrics = T)
nzvars <- nzvs[nzvs$nzv==T,0]

#Remove near zero variance predictors from train and test sets
smallTrain <- training[,!colnames(training) %in% rownames(nzvars)]
smallTest <- testing[,!colnames(testing) %in% rownames(nzvars)]
```

5.

Find predictors with large amount of NA values

```
x <- array()
for(i in 1:ncol(smallTrain)){
        x[i] <- sum(is.na(smallTrain[,i]))
}
print(x)
```

```
##   [1]     0     0     0     0     0     0     0     0     0     0 14420
##  [12] 14420 14420 14420 14420 14420 14420 14420 14420 14420 14420 14420
##  [23] 14420 14420 14420 14420     0     0     0     0     0     0     0
##  [34]     0     0     0     0     0     0 14420     0     0     0     0
##  [45]     0     0     0     0     0 14420 14420 14420 14420 14420 14420
##  [56] 14420 14420 14420     0     0     0 14420 14420 14420 14420 14420
##  [67] 14420     0 14420 14420 14420 14420 14420 14420 14420 14420 14420
##  [78] 14420     0     0     0     0     0     0     0     0     0     0
##  [89]     0     0 14420 14420 14420     0 14420     0     0     0     0
## [100]     0     0     0     0     0     0
```

```
table(x)
```

```
## x
##     0 14420
##    59    46
```

- The distribution of NA values across predictors with any NA value is skewed where every column with any NA value has **97.98%** of the total values missing, so we forego establishing a percentage NA threshold, and instead simply eliminate the predictors with any NA value at all.

6.

Remove Predictors with NA values

```
#subset train and test sets by rule above eliminating NA columns
smallerTrain <- smallTrain[,colSums(is.na(smallTrain)) == 0]
smallerTest <- smallTest[,colSums(is.na(smallTest)) == 0]
```

7.

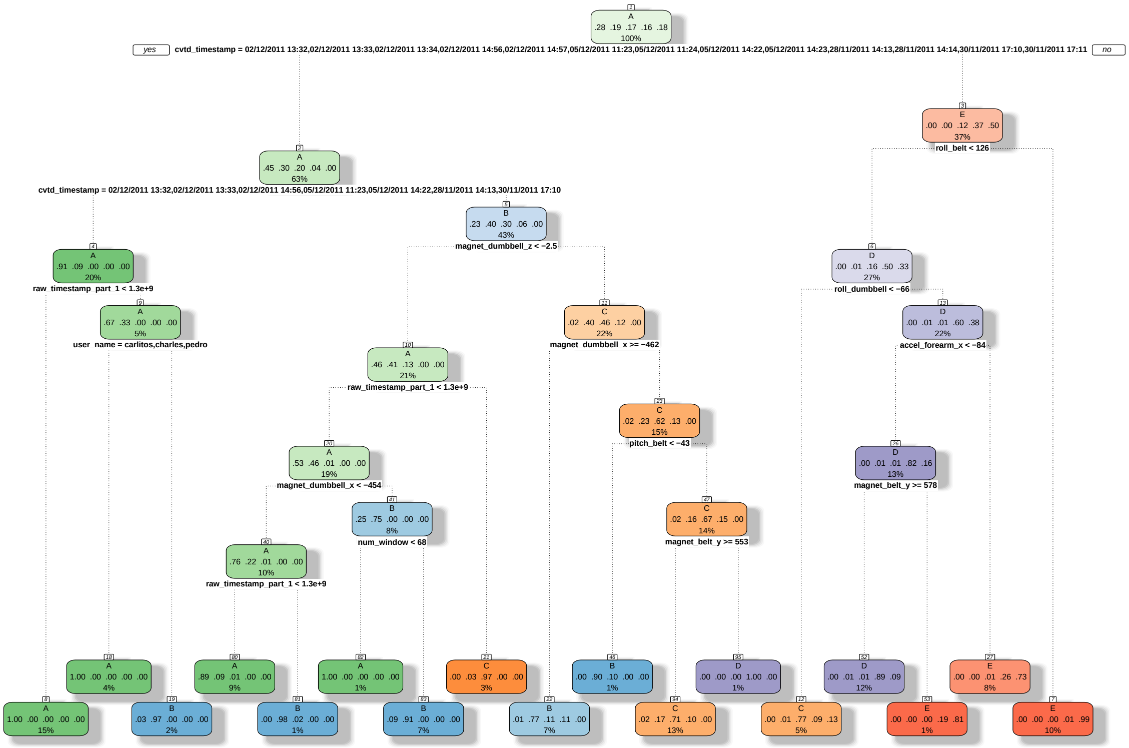
Final Data Cleaning Steps

```
#get rid of id number which is duplicate of row index

sTrain <- smallerTrain[,-1]
sTest <- smallerTest[,-1]
```

8.

Prediction with Decision Trees

```
#--------------
# Decision Tree
#--------------
set.seed(867)

tree <- rpart(classe ~ ., data = sTrain, method = 'class')
fancyRpartPlot(tree)
```



Rattle 2017−Mar−14 12:22:49 NNK

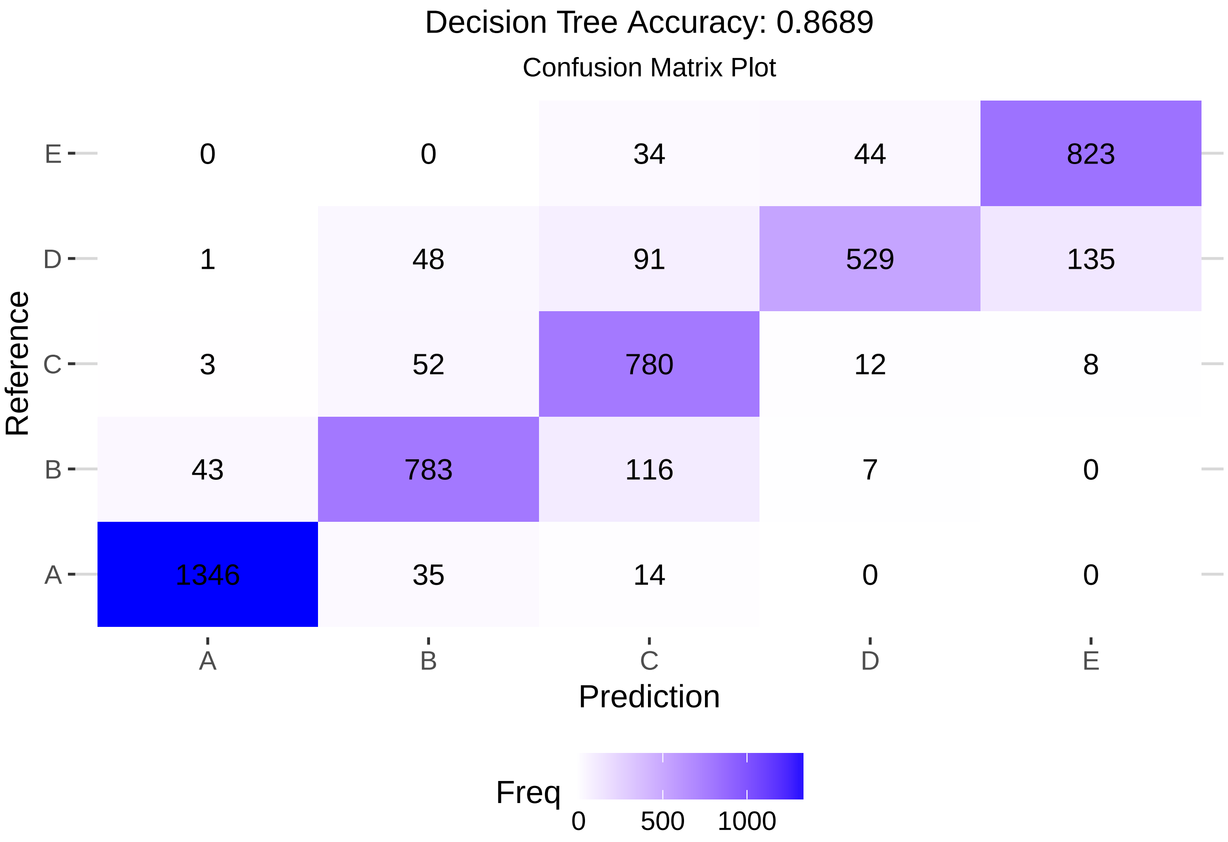
```
#Predict using tree
TreeFit <- predict(tree, sTest, type = 'class')
TreeResults <- confusionMatrix(TreeFit, testing$classe)

#Tree Accuracy
TreeResults$overall[1]
```

```
##  Accuracy
## 0.8688825
```

9.

Decision Tree Results

```
#Display confusion matrix results
tcm <- as.data.frame(TreeResults$table)

ggplot(tcm, aes(Prediction, Reference)) + geom_tile(aes(fill=Freq)) +
    geom_text(aes(label=digits(Freq,0))) +
    theme_hc()+
    scale_fill_gradient(low = "white", high = "blue") +
    ggtitle(label = paste("Decision Tree Accuracy:",round(TreeResults$overall['Accuracy'],4)),
            subtitle = "Confusion Matrix Plot") +
    theme(plot.title = element_text(hjust = 0.5, size = 12),
          plot.subtitle = element_text(hjust = 0.5, size = 10))
```


Decision Tree Accuracy: 0.8689

Confusion Matrix Plot

Reference	A	B	C	D	E
E	0	0	34	44	823
D	1	48	91	529	135
C	3	52	780	12	8
B	43	783	116	7	0
A	1346	35	14	0	0

10.

Prediction with Random Forests

```
set.seed(867)

forest <- train(classe ~ ., data = sTrain, method = 'rf')
```

```
## Loading required package: randomForest

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

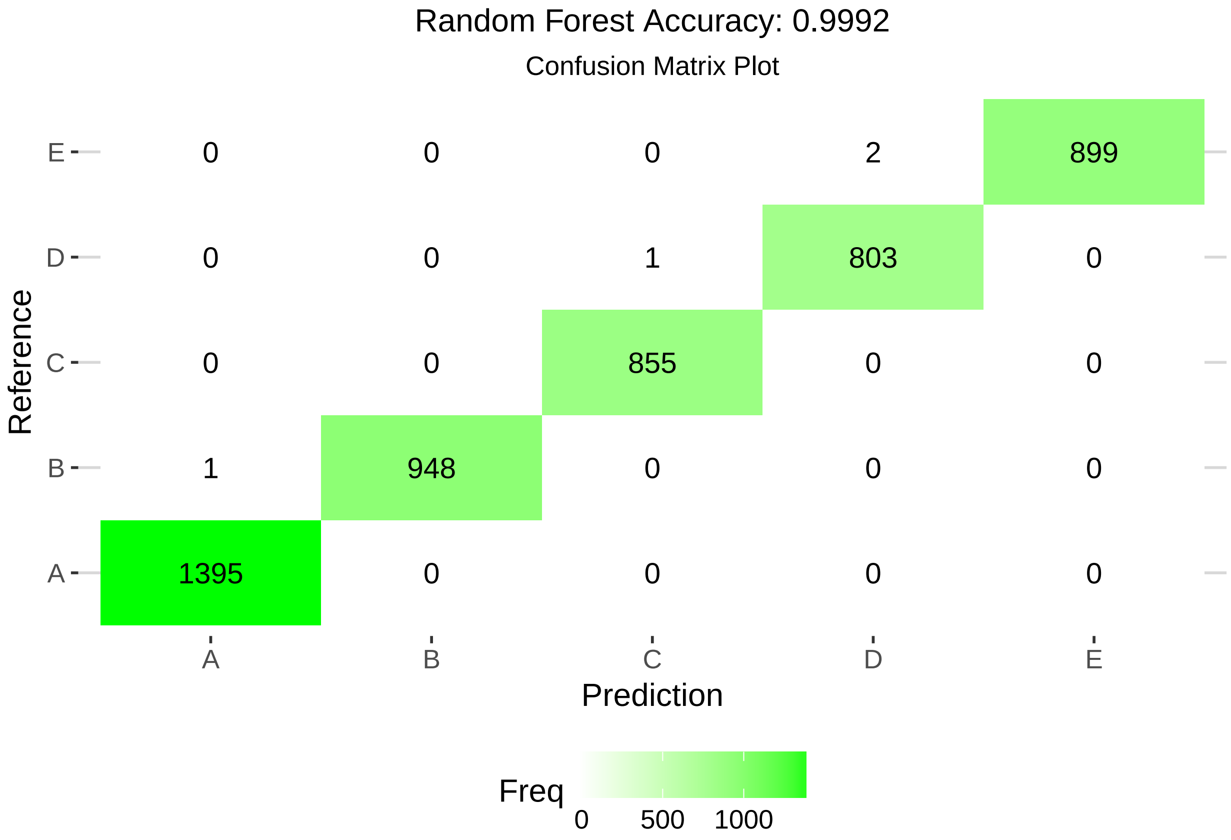
```
forestFit <- predict(forest, sTest)
forestResults <- confusionMatrix(forestFit, testing$classe)
forestResults$overall[1]
```

```
##  Accuracy
## 0.9991843
```

11.

Random Forest Results

```
fcm <- as.data.frame(forestResults$table)
ggplot(fcm, aes(Prediction, Reference)) + geom_tile(aes(fill=Freq)) +
    geom_text(aes(label=digits(Freq,0))) +
    theme_hc()+
    scale_fill_gradient(low = "white", high = "green") +
    ggtitle(label = paste("Random Forest Accuracy:",round(forestResults$overall['Accuracy'],4)),
            subtitle = "Confusion Matrix Plot") +
    theme(plot.title = element_text(hjust = 0.5, size = 12),
          plot.subtitle = element_text(hjust = 0.5, size = 10))
```



The best fit comes from Random Forests with an accuracy of 99.92%.

12.

Apply Random Forest model to test set

```
predict(forest, testing)[5]
```

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