Machine Learning Course Project

Jad Riachi

June 26, 2016

Loading packages used

```
library(caret)
```

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

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

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.3.1
```

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

```
## Warning: package 'rpart.plot' was built under R version 3.3.1
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.3.1
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
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
```

Read the training and the testing data and check the dimensions of each.

The code below requires manual download of the data:

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
(https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
(https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

```
setwd("~/Machine Learning/Course Project")
pml_training <- read.csv("pml-training.csv")
pml_testing <- read.csv("pml-testing.csv")
```

Exploring the Data

```
dim(pml_training)
```

```
## [1] 19622   160
```

```
dim(pml_testing)
```

```
## [1]  20 160
```

```
summary(pml_training$classe)
```

```
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

pml-training has 19622 observations of 160 variables and pml-testing set has 20 observations of 160 variables. The data will be be used to predict the manner of exercise or the ‘classe’ variable which is split into ‘A’, ‘B’, ‘C’, ‘D’, and ‘E’ factors.

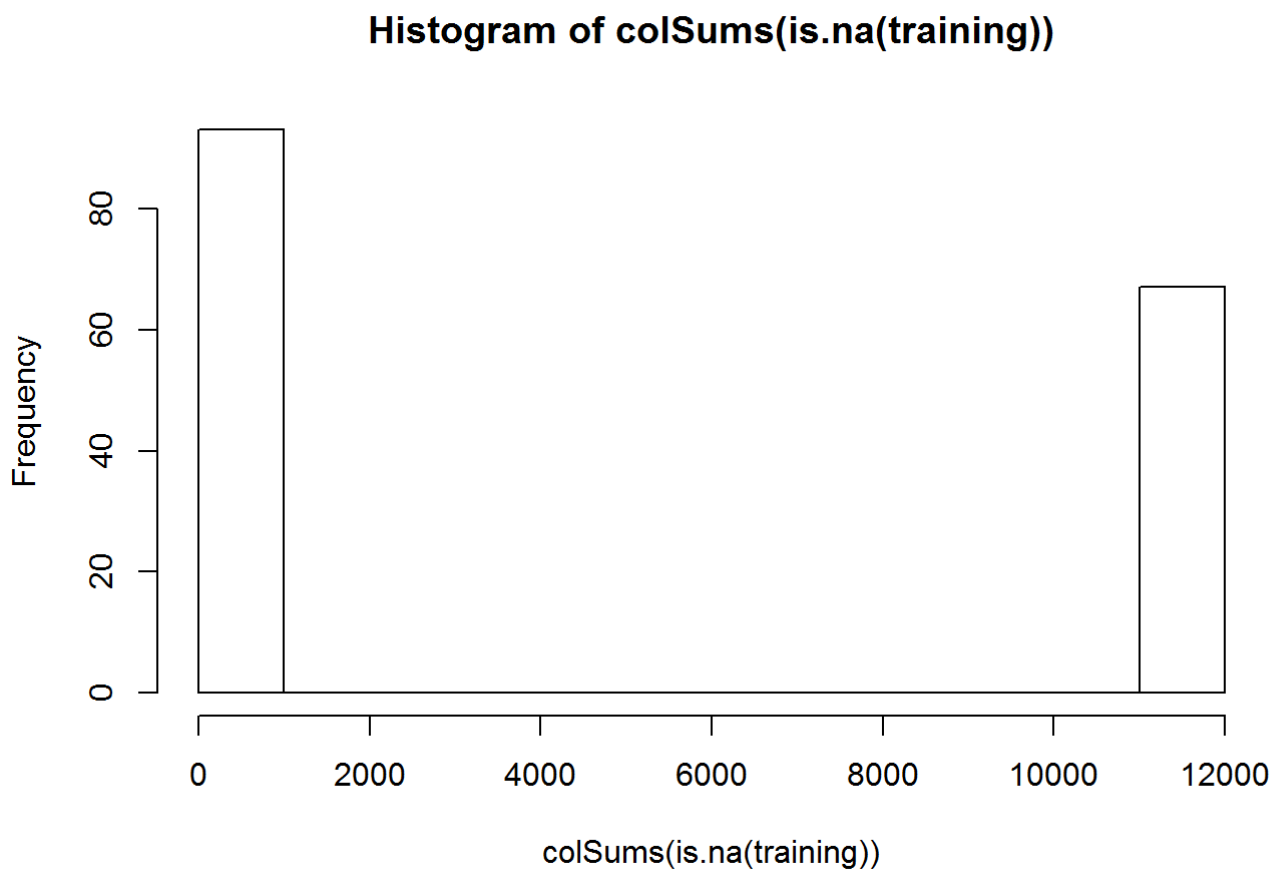
Partitioning our data

```
set.seed(303)
inTrain = createDataPartition(pml_training$classe, p = .6)[[1]]
training = pml_training[ inTrain,]
testing = pml_training[-inTrain,]
```

Exploring and Cleaing the training data

Find out how many of the varaibles have many missing values

```
hist(colSums(is.na(training)))
```



From the plot we notice that all variables either have a high ratio of NA values or a very low one. We therefore remove all variables with a high NA ratio from the training set

```
NAcount <- as.vector(colSums(is.na(training)))
NAindex <- NAcount > 10000
training <- training[,NAindex == FALSE]
```

With 93 variables reamining we will check for near zero variables or variables with very little variability that will not be useful for prediction.

```
nrzero <- nearZeroVar(training, saveMetrics = TRUE)
table(nrzero$nzv)
```

```
##
## FALSE  TRUE
##    59    34
```

34 of the remaining variables are classified as near zero variables and we remove them from our data as well.

```
training <- training[,nrzero$nzv == FALSE]
```

We will also remove the first 5 variables as they are not variables to build prediction models upon given that they are: - the index 'x' - 'user_name' - timestamps 'raw_timestamp_part_1' and 'raw_timestamp_part_2' and 'cvtd_timestamp'

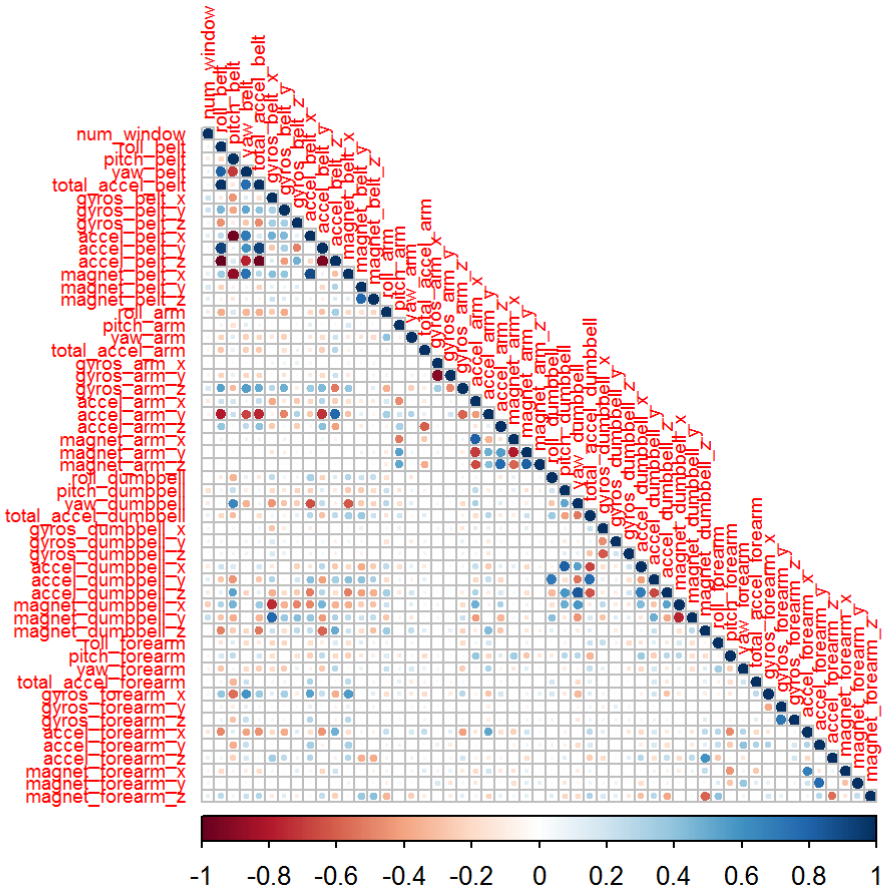
```
training <- training[,6:59]
```

We are left with 54 variables upon which we will build our prediction model.

Preprocessing

To determine the extent that the variables are correlated with each other we plot them on a correlation plot

```
cordf <- cor(training[,-54])
corrplot(cordf, type = "lower", tl.cex = 0.6)
```



The darker colors indicate which variables are highly correlated. Since these are few we will not conduct any further preprocessing on out data.

Prediciton

```
set.seed(313)
```

Decision Trees

```
modtree <- rpart(classe~., data = training, method = "class")

predtree <- predict(modtree, testing, type = "class")

confusionMatrix(testing$classe, predtree)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2017  123   24   38   30
##          B  298  916   97   86  121
##          C   20  146 1033   72   97
##          D  148  123  183  755   77
##          E  102  167  121  109  943
##
## Overall Statistics
##
##                Accuracy : 0.7219
##                  95% CI : (0.7118, 0.7318)
##     No Information Rate : 0.3295
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.646
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7803   0.6210   0.7085  0.71226   0.7437
## Specificity            0.9591   0.9055   0.9476  0.92175   0.9241
## Pos Pred Value         0.9037   0.6034   0.7551  0.58709   0.6540
## Neg Pred Value         0.8988   0.9117   0.9344  0.95351   0.9493
## Prevalence             0.3295   0.1880   0.1858  0.13510   0.1616
## Detection Rate         0.2571   0.1167   0.1317  0.09623   0.1202
## Detection Prevalence   0.2845   0.1935   0.1744  0.16391   0.1838
## Balanced Accuracy      0.8697   0.7633   0.8280  0.81701   0.8339
```

To plot the tree execute fancyRpartPlot(modtree). Plot not shown here due to size.

Random Forests

```
modrf <- randomForest(classe~., data = training)

predrf <- predict(modrf, testing)

confusionMatrix(testing$classe, predrf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2231    1    0    0    0
##          B   10 1508    0    0    0
##          C    0    9 1358    1    0
##          D    0    0    5 1280    1
##          E    0    0    0    3 1439
##
## Overall Statistics
##
##                Accuracy : 0.9962
##                  95% CI : (0.9945, 0.9974)
##     No Information Rate : 0.2856
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9952
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9955   0.9934   0.9963   0.9969   0.9993
## Specificity            0.9998   0.9984   0.9985   0.9991   0.9995
## Pos Pred Value         0.9996   0.9934   0.9927   0.9953   0.9979
## Neg Pred Value         0.9982   0.9984   0.9992   0.9994   0.9998
## Prevalence             0.2856   0.1935   0.1737   0.1637   0.1835
## Detection Rate         0.2843   0.1922   0.1731   0.1631   0.1834
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9977   0.9959   0.9974   0.9980   0.9994
```

The accuracy of the model built using the random forest has an accuracy of 99.62% as opposed to 72.19% accuracy of the decision trees method.

We will therefore use the random forest model to predict our 20 data points from the pml_testing data that will be submitted.

```
predrf <- predict(modrf, pml_testing)
predrf
```

```
##  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
```

Citation

Data downloaded from the following source:

Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6.

Read more: http://groupware.les.inf.puc-rio.br/har#ixzz4Cfvudt92 (http://groupware.les.inf.puc-rio.br/har#ixzz4Cfvudt92)