Course Project Machine Learning

Deborah Castillo

17/10/2017

First, I will load the data and corresponding packages that will be used.

```
testing <- read.csv(file = "pml-testing.csv", header = TRUE)
training <- read.csv(file = "pml-training.csv", header = TRUE)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.5
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.2.4
```

To use cross validation, it is neccesary to split the training dataframe again. Conventionally, the split is made 60% into the training set and 40 into the new testing set.

```
inTrain <- createDataPartition(y = training$classe, p =0.6, list = FALSE)
trains <- training[inTrain, ]
tests <- testing[-inTrain, ]
```

Next, the data needs to be cleaned. For this, I will only extract the variables I consider relevant for the analysis. This is done because I want to use the random forest method for prediction, and it takes a lot of RAM memory, so I want to keep only the neccesary variables.

```
relvar <- names(trains) %in% c("classe", "accel_arm_x", "accel_arm_y", "accel_arm_z", "accel_belt_x", "a
trains <- trains[relvar]
tests <- tests[relvar]
tests$problem_id <- sample(trains$classe, size = nrow(tests), replace = TRUE)
tests$problem_id <- factor(tests$problem_id)
colnames(tests)[53] <- "classe"
```

Following, a predictive model will be made using the ranom forests package, as mentioned before

```
library(rpart)
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
```

```
fit1 <- rpart(classe ~ . , data = trains)
fit2 <- randomForest(classe ~ . , data = trains)
fit2
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = trains)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.6%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3345    2    0    0    1 0.0008960573
## B   11 2261    7    0    0 0.0078982010
## C    0   11 2038    5    0 0.0077896787
## D    0    0   24 1905    1 0.0129533679
## E    0    0    2    7 2156 0.0041570439
```

As we can see above, the OOB is extremely high, this is why cross-validation is so important in random forest.

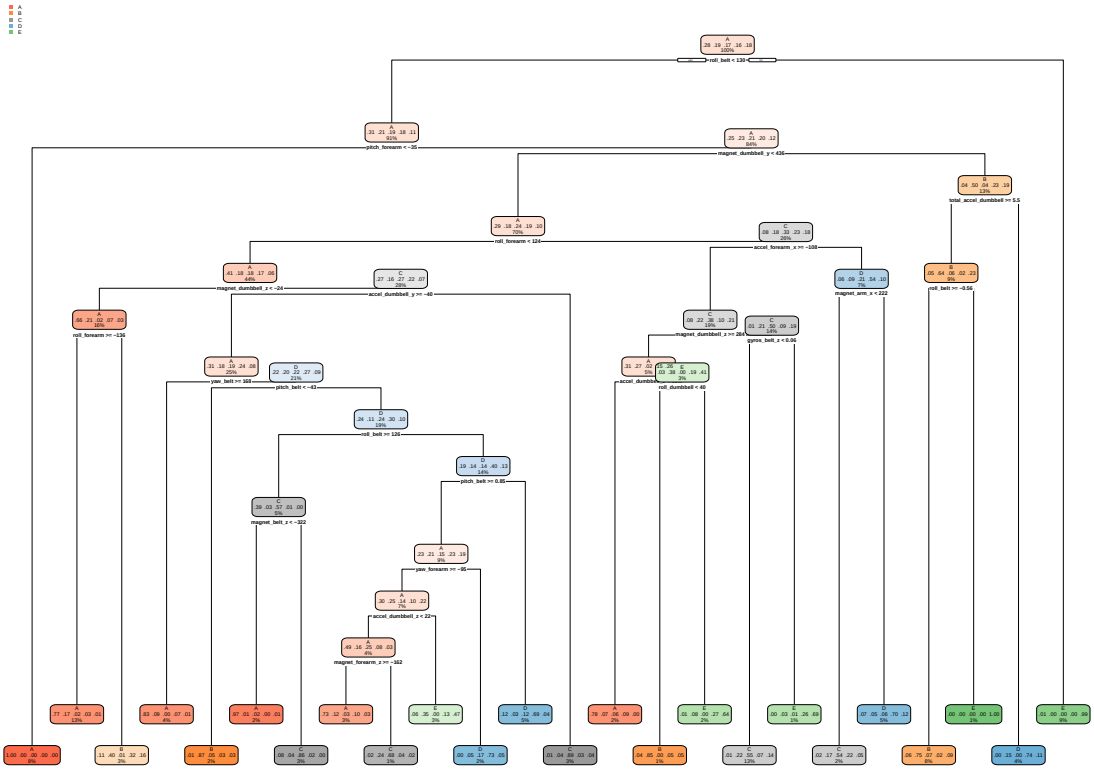
I will graph the prior, and then do some cross validation

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

```
## Warning: package 'rpart.plot' was built under R version 3.2.5
```

```
rpart.plot(fit1)
```

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



```
prediction <- predict(fit1, tests, type = "class")
confusionMatrix(tests$classe, prediction)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction A B C D E
##          A 0 0 0 0 1
##          B 0 0 1 0 0
##          C 0 0 1 0 0
##          D 2 1 1 0 0
##          E 0 1 0 0 0
##
## Overall Statistics
##
##                Accuracy : 0.125
##                  95% CI : (0.0032, 0.5265)
##     No Information Rate : 0.375
##     P-Value [Acc > NIR] : 0.9767
##
##                   Kappa : 0
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.0000   0.0000   0.3333       NA   0.0000
## Specificity            0.8333   0.8333   1.0000      0.5   0.8571
## Pos Pred Value         0.0000   0.0000   1.0000       NA   0.0000
## Neg Pred Value         0.7143   0.7143   0.7143       NA   0.8571
```

```
## Prevalence             0.2500   0.2500   0.3750      0.0   0.1250
## Detection Rate         0.0000   0.0000   0.1250      0.0   0.0000
## Detection Prevalence   0.1250   0.1250   0.1250      0.5   0.1250
## Balanced Accuracy      0.4167   0.4167   0.6667       NA   0.4286
```

As we can see above, I only obtained a 22% accuracy in the rpart method. This might be because the model does not fit the data correcly.

Final prediction

```
finalpredict <- predict(fit1, testing, type = "class")
print(finalpredict)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  C  A  C  D  A  C  D  A  A  A  C  C  B  A  C  E  A  B  B  B
## Levels: A B C D E
```