Prediction Assignment Writeup

Marcello Remedia

Summary

The goal of the project is to predict the manner in which the people analysed did the exercise. In particular, a training set with 19622 observations was provided and the best method to predict the outcome is to be chosen for the test set of 20 observations. KNN, decision trees, random forest and boosting were chosen as methods to be tested splitting the training data in two (choosing 70% of the observations as training and 30% as test). Cross-validation is applied where possible. The method providing the best accuracy was found to be random forest, which was finally used to obtain the final predictions.

Preprocessing

As a first step, the libraries needed are loaded. Same goes for the training and the test sets. The test set in particular has a lot of column with only NA values. These columns are deleted in both the test and the train sets.

```
library(caret)
```

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

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

```
library(rpart)
library(randomForest)
```

```
## randomForest 4.6-14
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
library(rpart.plot)
set.seed(123)

train <- read.csv('pml-training.csv')
test <- read.csv('pml-testing.csv')
test <- test[, -c(1:7,dim(test)[2])]
test <- test[, colSums(is.na(test)) == 0]
train <- train[, c(names(test), "classe")]
train$classe <- factor(train$classe)
```

Next, the training data is split into two to test every single method on data for which we actually have results.

```
indexes <- createDataPartition(train$classe, p=0.7, list=FALSE)
train_tr <- train[indexes,]
train_te <- train[-indexes,]
```

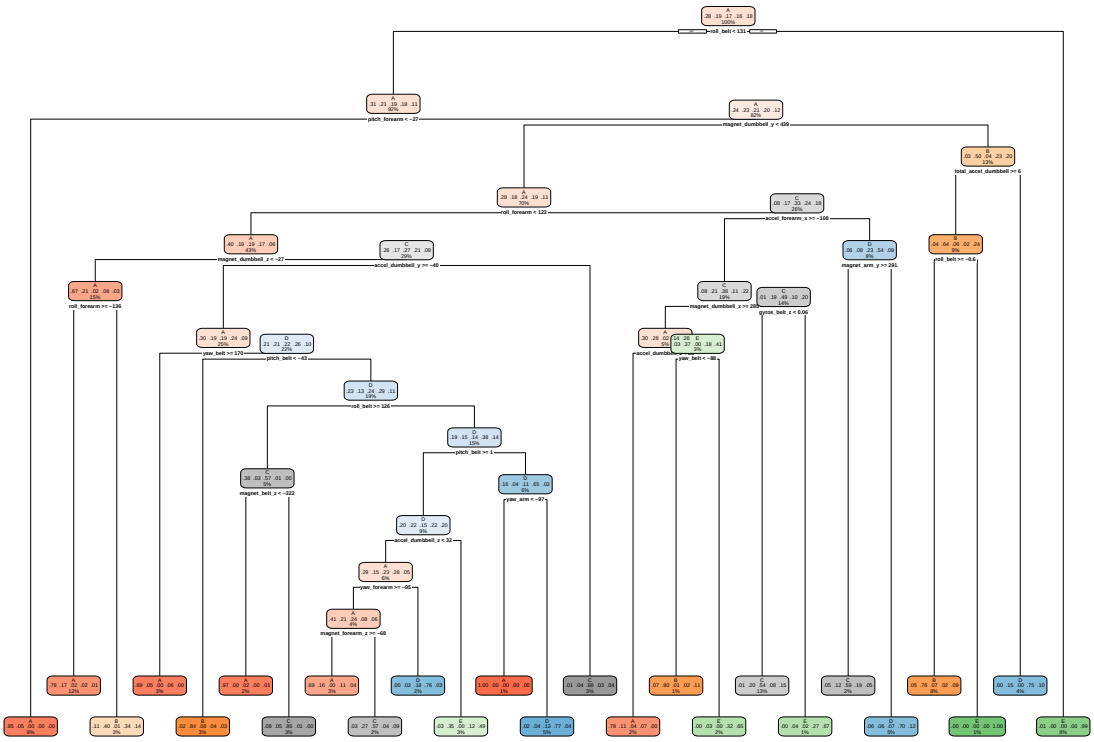
KNN Model

```
knn <- train(classe ~ ., data = train_tr, method = "knn", trControl = trainControl(method = "repeatedcv
```

Decision Trees

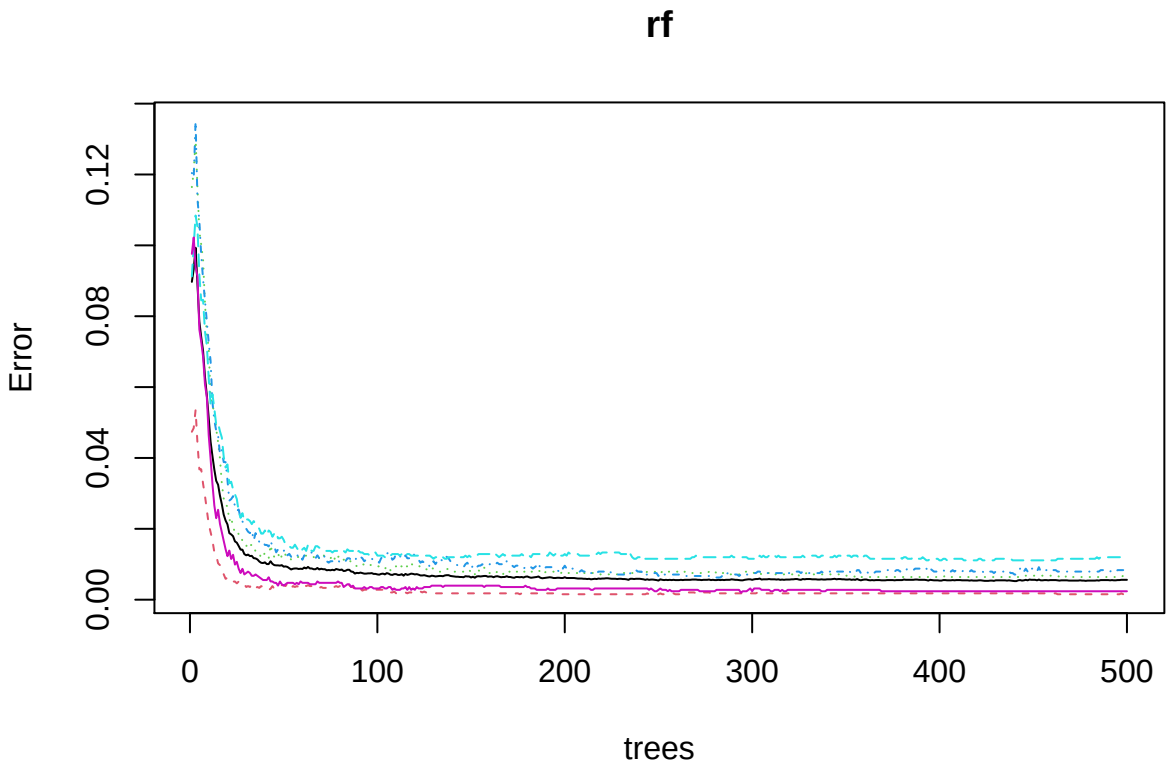
```
dectree <- rpart(classe ~ ., data=train_tr, method="class")
rpart.plot(dectree)
```

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



Random Forest

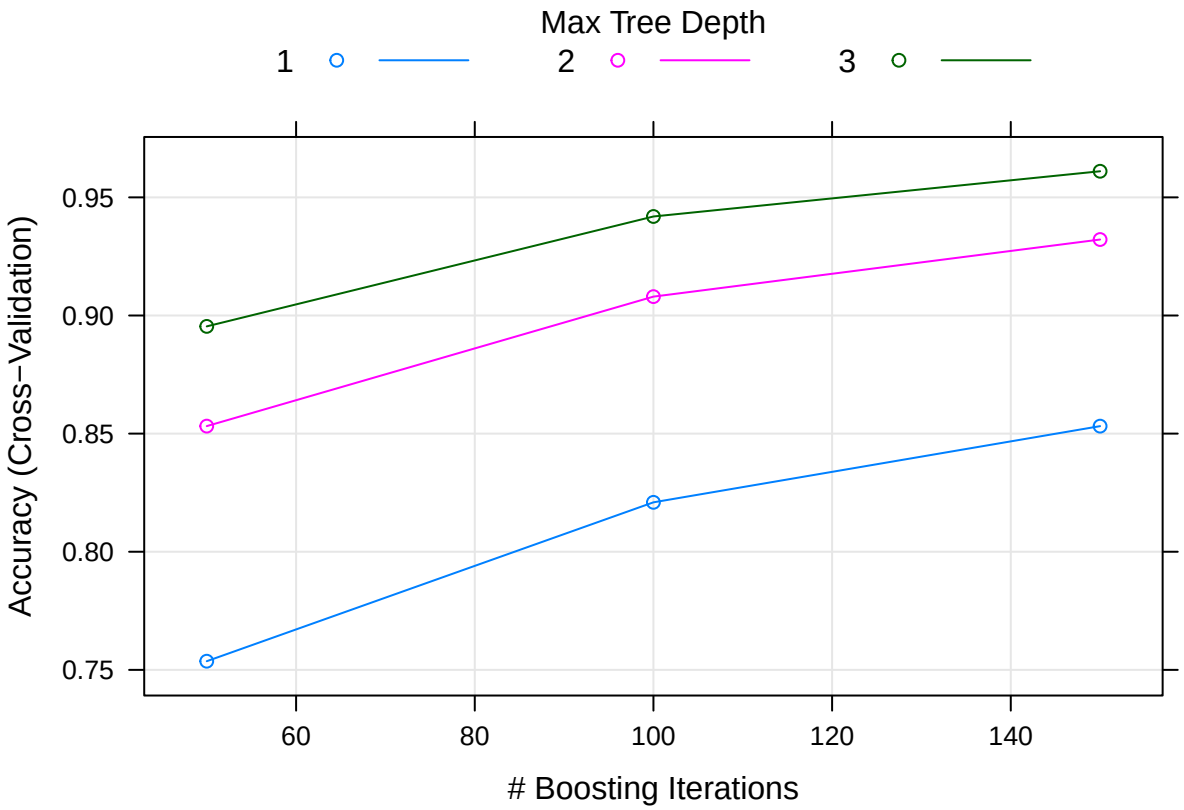
```
rf <- randomForest(classe ~ ., data = train_tr, method = "rf", trControl = trainControl(method = "cv", 
plot(rf)
```



plot(knn)

Boosting

```
gbm <- train(classe ~ ., data = train_tr, method = "gbm", verbose = F, trControl = trainControl(method =
plot(gbm)
```



Predictions

First, a dataframe containing the models and the accuracies is initialised.

```
acc_mat <- data.frame(model = c("knn", "decision tree", "random forest", "boosting"), accuracy = c(0,0,
```

Then, predictions for all models are performed. For each one of them, the confusion matrix is built, and from it the accuracy is taked and inserted into the dataframe.

```
knn_pred <- predict(knn, train_te)
c <- confusionMatrix(knn_pred, train_te$classe)
acc_mat[1,2] <- c$overall[1]
dectree_pred <- predict(dectree, train_te, type = "class")
c <- confusionMatrix(dectree_pred, train_te$classe)
acc_mat[2,2] <- c$overall[1]
rf_pred <- predict(rf, train_te, type = "class")
c <- confusionMatrix(rf_pred, train_te$classe)
acc_mat[3,2] <- c$overall[1]
gbm_pred <- predict(gbm, train_te)
c <- confusionMatrix(gbm_pred, train_te$classe)
acc_mat[4,2] <- c$overall[1]
```

This is the final dataframe:

```
acc_mat
```

```
##           model  accuracy
## 1           knn 0.9619371
## 2 decision tree 0.7573492
## 3 random forest 0.9949023
## 4      boosting 0.9615973
```

Random Forest is therefore chosen and these are the prediction on the final test set:

```
prediction <- predict(rf, test)
prediction
```

```
##  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
```