Machine Learning Course Assignment

Introduction

One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, the goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants.

The goal of the project is to predict the manner in which they did the exercise. This is the “classe” variable in the training set. May use any of the other variables to predict with and should create a report describing how the model is built, how cross validation is performed, what the expected out of sample error is, and why the choices were done. Lastly, use the prediction model to predict 20 different test cases.

The data available for this project are original Training Data (pml-training.csv) and Testing Data (pml-testing.csv). The Training Data is used for modeling, prediction and validation. The Testing Data contains 20 different test cases on which the final prediction model will perform prediction.

Modeling

Our outcome variable is classe, a factor variable with 5 levels. Prediction evaluations will be based on maximizing the accuracy and minimizing the out-of-sample error. Two models will be tested using decision tree and random forest algorithms. The model with the highest accuracy will be chosen as our final model.

Cross-validation will be performed by partitioning our Training data randomly without replacement into 2 subsets: “trainingset” data (80% of the original Training data) and “testingset” data (the rest 20%). Our models will be fitted on the “trainingset” data set, and tested on the “testingset” data set. Once the most accurate model is chosen, it will be tested on the original Testing data.

Results

1. loading required packages

```
#install.packages("caret")
#install.packages("randomForest")
#install.packages("rpart")
#install.packages("rpart.plot")
#install.packages('e1071', dependencies=TRUE) # Package related to confusionMatr
ix

library(caret)
```

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

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

```
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(rpart)
library(rpart.plot)
set.seed(88888888)
```

2. loading and processing data

```
training <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!", ""))
testing <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))

training<-training[,colSums(is.na(training)) == 0]
testing <-testing[,colSums(is.na(testing)) == 0]

training   <-training[,-c(1:7)]
testing <-testing[,-c(1:7)]

dim(training)
```

```
## [1] 19622    53
```

```
dim(testing)
```

```
## [1] 20 53
```

```
head(training$classe,30)
```

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

```
tail(training$classe,30)
```

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

```
#head(testing)
```

3. partitioning the training data into training set and testing set

```
samples <- createDataPartition(y=training$classe, p=0.80, list=FALSE)
trainingset <- training[samples, ]
testingset <- training[-samples, ]

dim(trainingset)
```

```
## [1] 15699    53
```

```
dim(testingset)
```

```
## [1] 3923   53
```

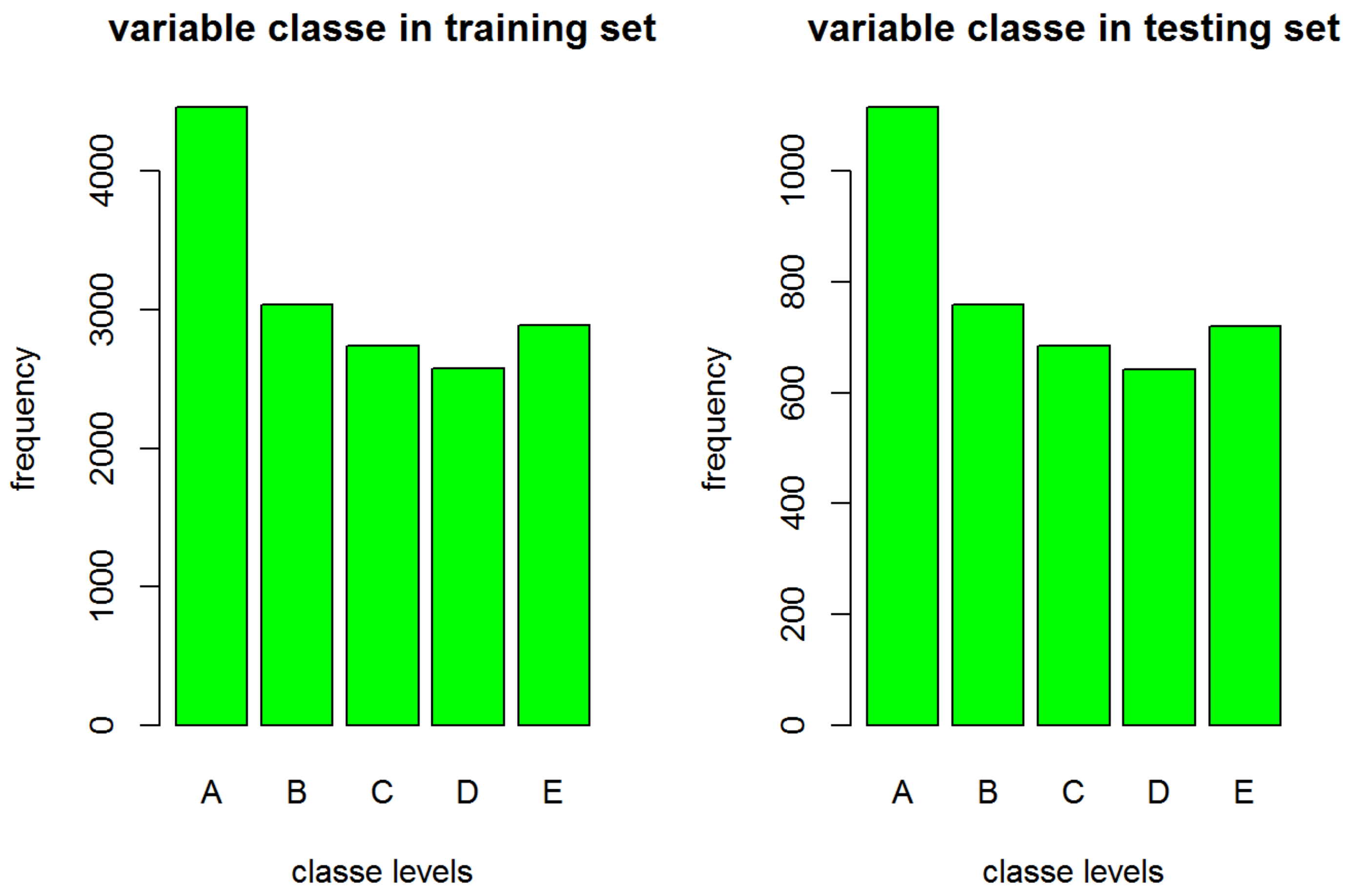
```
#head(trainingset)
#head(testingset)

dim(trainingset)[1]/dim(training)[1] # check percentage (80%) of the tranining s
et portion
```

```
## [1] 0.8000713
```

4. checking the two partition sets with plot

```
par(mfrow=c(1,2))
plot(trainingset$classe, col="green", main="variable classe in training set", xl
ab="classe levels", ylab="frequency")
plot(testingset$classe, col="green", main="variable classe in testing set", xlab
="classe levels", ylab="frequency")
```

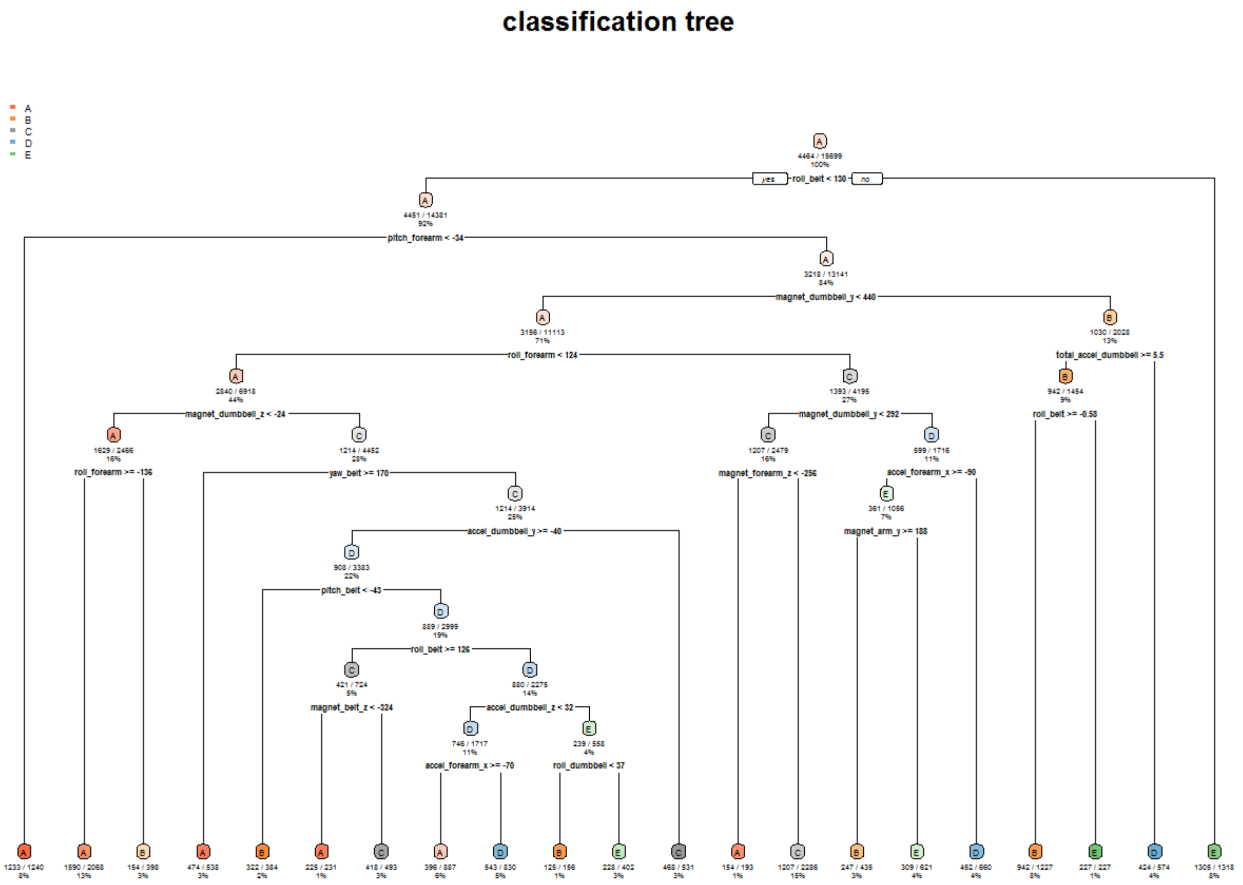


5. prediction model 1: Decision Tree method

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

pred1 <- predict(mod1, testingset, type = "class") # Predicting

rpart.plot(mod1, main="classification tree", extra=102, under=TRUE, faclen=0) #
Plot of decision tree
```



```
confusionMatrix(pred1, testingset$classe) # Test results on partition testing se
t
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   A   B   C   D   E
##          A 994 122  62  87  21
##          B  41 449  49  56  63
##          C  36  85 500 101  86
##          D  26  59  56 342  34
##          E  19  44  17  57 517
## 
## Overall Statistics
## 
##                Accuracy : 0.7142          
##                  95% CI : (0.6998, 0.7283)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
## 
##                   Kappa : 0.6365          
##  Mcnemar's Test P-Value : < 2.2e-16       
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8907   0.5916   0.7310  0.53188   0.7171
## Specificity            0.8960   0.9339   0.9049  0.94665   0.9572
## Pos Pred Value         0.7729   0.6824   0.6188  0.66151   0.7905
## Neg Pred Value         0.9537   0.9051   0.9409  0.91163   0.9376
## Prevalence             0.2845   0.1935   0.1744  0.16391   0.1838
## Detection Rate         0.2534   0.1145   0.1275  0.08718   0.1318
## Detection Prevalence   0.3278   0.1677   0.2060  0.13179   0.1667
## Balanced Accuracy      0.8933   0.7628   0.8180  0.73926   0.8371
```

6. prediction model 2: Random Forest method

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

pred2 <- predict(mod2, testingset, type = "class") # Predicting

confusionMatrix(pred2, testingset$classe) # Test results on partition testing se
t
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1114    5    0    0    0
##          B    2  751    4    0    0
##          C    0    3  679   10    0
##          D    0    0    1  633    0
##          E    0    0    0    0  721
## 
## Overall Statistics
## 
##                Accuracy : 0.9936
##                  95% CI : (0.9906, 0.9959)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9919
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9982   0.9895   0.9927   0.9844   1.0000
## Specificity            0.9982   0.9981   0.9960   0.9997   1.0000
## Pos Pred Value         0.9955   0.9921   0.9812   0.9984   1.0000
## Neg Pred Value         0.9993   0.9975   0.9985   0.9970   1.0000
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2840   0.1914   0.1731   0.1614   0.1838
## Detection Prevalence   0.2852   0.1930   0.1764   0.1616   0.1838
## Balanced Accuracy      0.9982   0.9938   0.9943   0.9921   1.0000
```

7. choosing the final model

Accuracy for Random Forest model was 0.994 compared to 0.714 for Decision Tree model. The random Forest model is chosen. The accuracy of the model is 0.994. The expected out-of-sample error is estimated at 0.6%. The expected out-of-sample error is calculated as 1 - accuracy for predictions made against the cross-validation set.

Note that Accuracy is the proportion of correct classified observation over the total sample in the “testingset” data set. The expected value of the out-of-sample error will correspond to the expected number of missclassified observations/total observations in the Test data set, which is the quantity: 1-accuracy found from the cross-validation data set.

8. final prediction results on original testing data

```
predfinal <- predict(mod2, testing, type="class")
predfinal
```

```
##  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
```