PMLWeek4

Pratik Patil

18/10/2020

##Synopsis

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks.
The goal of this project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

##Understanding the Data

The outcome variable is `classe`, a factor variable with 5 levels. For this data set, participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different fashions:

- Class A: exactly according to the specification
- Class B: throwing the elbows to the front
- Class C: lifting the dumbbell only halfway
- Class D: lowering the dumbbell only halfway
- Class E: throwing the hips to the front

##Loading Packages and Initializing Varialbes

```
#Data variables declaration
train.file   <- 'D:/downloads/pml-training.csv'
test.case.file <- 'D:/downloads/pml-testing.csv'
train.url    <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
test.case.url  <- 'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
#Directories
if (!file.exists("data")){
  dir.create("data")
}
if (!file.exists("data/submission")){
  dir.create("data/submission")
}
#R-Packages install check
IscaretInstalled <- require("caret")
```

```
## Loading required package: caret
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

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

```
if(!IscaretInstalled){
    install.packages("caret")
    library("caret")
    }
IsrandomForestInstalled <- require("randomForest")
```

```
## Loading required package: randomForest
```

```
## Warning: package 'randomForest' was built under R version 3.6.3
```

```
## 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
```

```
if(!IsrandomForestInstalled){
    install.packages("randomForest")
    library("randomForest")
    }
IsRpartInstalled <- require("rpart")
```

```
## Loading required package: rpart
```

```
if(!IsRpartInstalled){
    install.packages("rpart")
    library("rpart")
    }
IsRpartPlotInstalled <- require("rpart.plot")
```

```
## Loading required package: rpart.plot
```

```
## Warning: package 'rpart.plot' was built under R version 3.6.3
```

```
if(!IsRpartPlotInstalled){
    install.packages("rpart.plot")
    library("rpart.plot")
    }
# Setting seed for reproducability purpose
set.seed(9999)
```

Data Processing

Downloading and processing of data is done. Some transformation and cleaning will be performed, so that `NA` values are not considered. Irrelevant columns (columns 1 to 7) will be removed in the subset.

The `pml-training.csv` data is used to devise training and testing sets. The `pml-test.csv` data is used to predict and answer the 20 questions based on the trained model.

```
# Download data
download.file(train.url, train.file)
download.file(test.case.url,test.case.file )
# Clean data
train   <-read.csv(train.file, na.strings=c("NA","#DIV/0!", ""))
test <-read.csv(test.case.file , na.strings=c("NA", "#DIV/0!", ""))
train<-train[,colSums(is.na(train)) == 0]
test <-test[,colSums(is.na(test)) == 0]
# Subset data
train   <-train[,-c(1:7)]
test <-test[,-c(1:7)]
```

Cross-validation

Cross-validation will be performed by splitting the training data in training (75%) and testing (25%) data.

```
subSams <- createDataPartition(y=train$classe, p=0.75, list=FALSE)
subTrain <- train[subSams, ]
subTest <- train[-subSams, ]
```

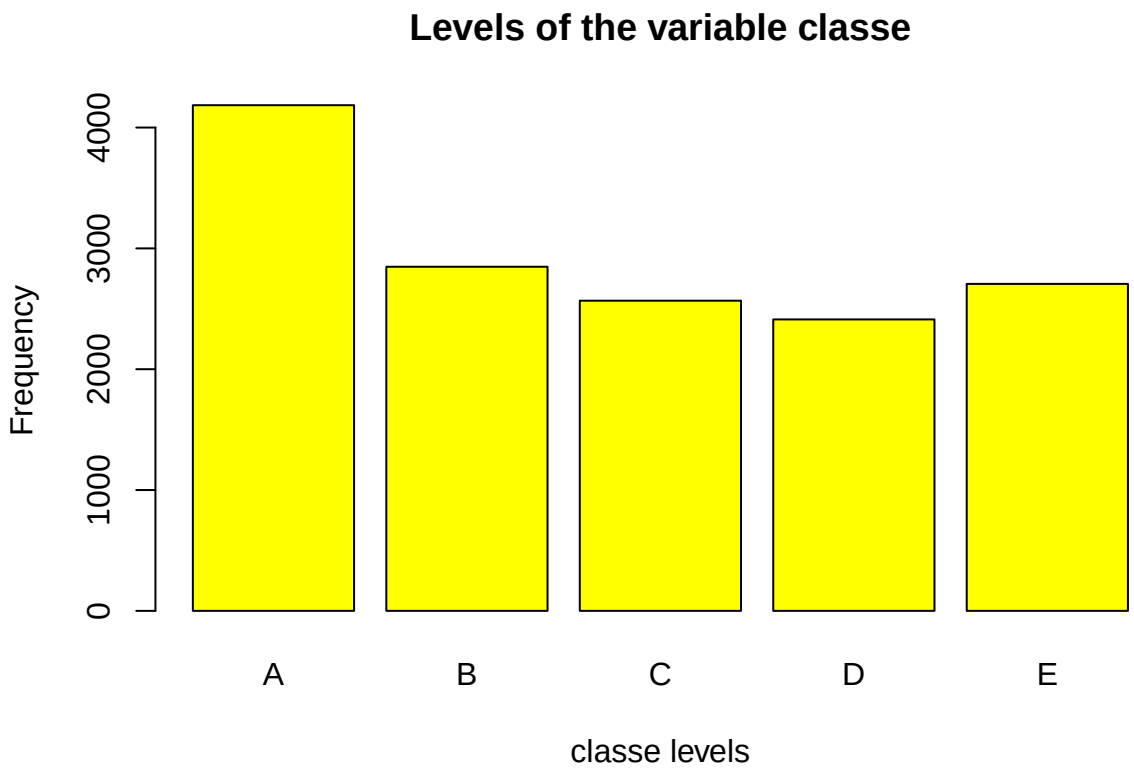
Expected out-of-sample error

The expected out-of-sample error will correspond to the quantity: 1-accuracy in the cross-validation data. Accuracy is the proportion of correct classified observation over the total sample in the subTesting data set. Expected accuracy is the expected accuracy in the out-of-sample data set (i.e. original testing data set). Thus, 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.

Exploratory analysis

The variable `classe` contains 5 levels. The plot of the outcome variable shows the frequency of each levels in the subTraining data.

```r
plot(subTrain$classe, col="yellow", main="Levels of the variable classe", xlab="classe levels", ylab="Fr
```



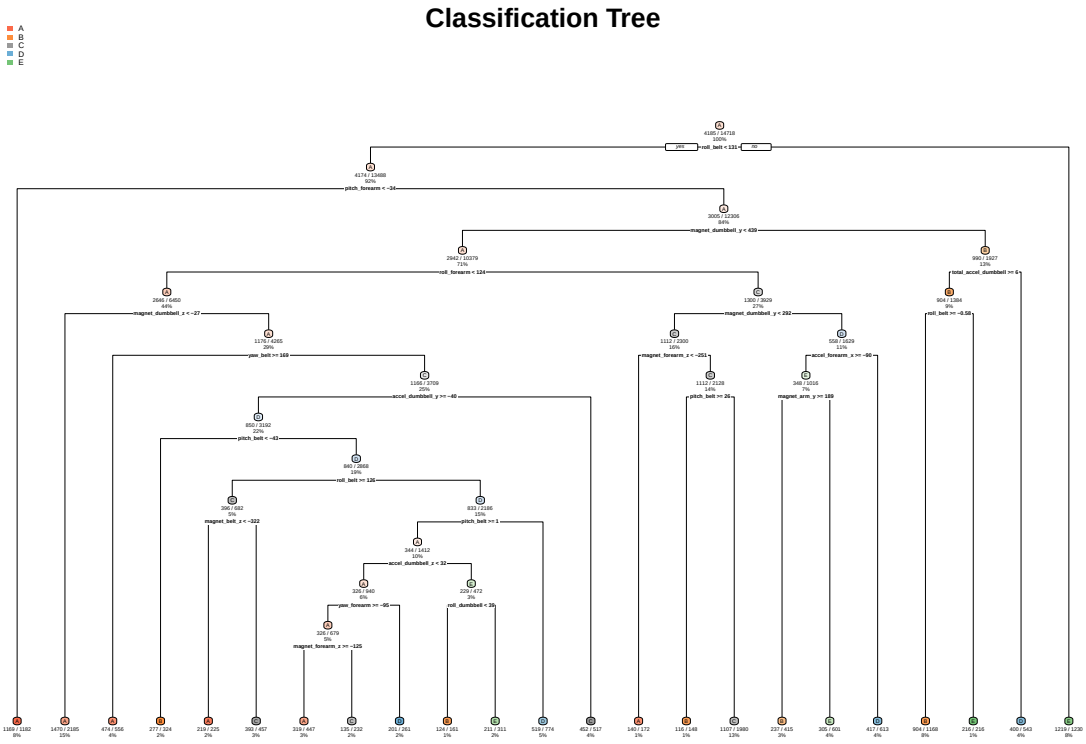
The plot above shows that Level A is the most frequent classe. D appears to be the least frequent classe.

Prediction models

In this section a decision tree and random forest will be applied to the data.

Decision tree

```r
# Fit model
modFDT <- rpart(classe ~ ., data=subTrain, method="class")
# Perform prediction
predDT <- predict(modFDT, subTest, type = "class")
# Plot result
rpart.plot(modFDT, main="Classification Tree", extra=102, under=TRUE, faclen=0)
```



Following confusion matrix shows the errors of the prediction algorithm.

```
confusionMatrix(predDT, subTest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1247  212   23   83   30
##          B   32  530   73   23   73
##          C   35   96  695  112  121
##          D   60   66   46  532   46
##          E   21   45   18   54  631
##
## Overall Statistics
##
##                Accuracy : 0.7412
##                  95% CI : (0.7287, 0.7534)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6712
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8939   0.5585   0.8129   0.6617   0.7003
## Specificity            0.9008   0.9492   0.9101   0.9468   0.9655
## Pos Pred Value         0.7818   0.7250   0.6563   0.7093   0.8205
## Neg Pred Value         0.9553   0.8996   0.9584   0.9345   0.9347
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2543   0.1081   0.1417   0.1085   0.1287
## Detection Prevalence   0.3252   0.1491   0.2159   0.1529   0.1568
## Balanced Accuracy      0.8974   0.7538   0.8615   0.8043   0.8329
```

Random forest

```
# Fit model
modFRF <- randomForest(classe ~ ., data=subTrain, method="class")
# Perform prediction
predRF <- predict(modFRF, subTest, type = "class")
```

Following confusion matrix shows the errors of the prediction algorithm.

```
confusionMatrix(predRF, subTest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1393    5    0    0    0
##          B    2  943    2    0    0
##          C    0    1  853    8    0
##          D    0    0    0  795    2
##          E    0    0    0    1  899
##
## Overall Statistics
##
##                Accuracy : 0.9957
##                  95% CI : (0.9935, 0.9973)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9946
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9986   0.9937   0.9977   0.9888   0.9978
## Specificity            0.9986   0.9990   0.9978   0.9995   0.9998
## Pos Pred Value         0.9964   0.9958   0.9896   0.9975   0.9989
## Neg Pred Value         0.9994   0.9985   0.9995   0.9978   0.9995
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
```

```
## Detection Rate         0.2841   0.1923   0.1739   0.1621   0.1833
## Detection Prevalence   0.2851   0.1931   0.1758   0.1625   0.1835
## Balanced Accuracy      0.9986   0.9963   0.9977   0.9942   0.9988
```

Conclusion

Result

The confusion matrices show, that the Random Forest algorithm performens better than decision trees. The accuracy for the Random Forest model was 0.995 (95% CI: (0.993, 0.997)) compared to 0.739 (95% CI: (0.727, 0.752)) for Decision Tree model. The random Forest model is choosen.

Expected out-of-sample error

The expected out-of-sample error is estimated at 0.005, or 0.5%. The expected out-of-sample error is calculated as 1 - accuracy for predictions made against the cross-validation set. Our Test data set comprises 20 cases. With an accuracy above 99% on our cross-validation data, we can expect that very few, or none, of the test samples will be missclassified.

Submission

In this section the files for the project submission are generated using the random forest algorithm on the testing data.

```
# Perform prediction
predSub <- predict(modFRF, test, type="class")
predSub
```

```
##  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
```

```
# Write files for submission
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("./data/submission/problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
pml_write_files(predSub)
```