Practical Machine Learning

Prediction Assignment - Week 4

Olga Koroleva

Abstract

The goal of this project is to predict the manner in which people exercise based on accelerometers on the belt, forearm, arm, and dumbbell of 6 participants from the Weight Lifting Exercise Dataset using different machine learning algorithms.

Six participants were asked to perform barbell lifts correcty and incorrectly in five different manners wearing fitness trackers like Jawbone Up, Nike FuelBand, and Fitbit in this dataset. The data gained from these devices is used to train the models.

Data Sources

The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har

Data Import & Transformation

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:

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

The initial configuration consists of loading some required packages and initializing some variables.

```
library(caret)
```

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

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

```
library(randomForest)
```

```
## randomForest 4.7-1.2
```

```
## 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 for reproducability
set.seed(9999)

# Load 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]

# Subset data
training   <-training[,-c(1:7)]
testing <-testing[,-c(1:7)]
```

Cross-Validation

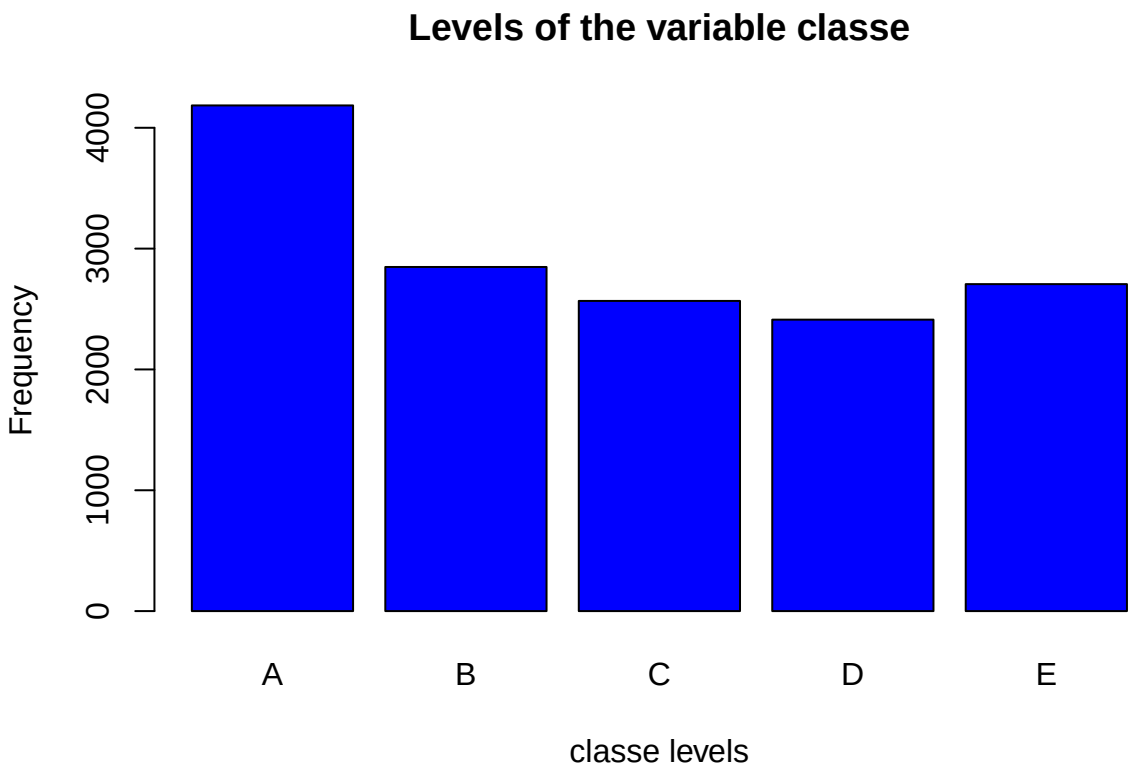
In this section cross-validation will be performed by splitting the training data in training (75%) and testing (25%) data.

```
subSamples <- createDataPartition(y=training$classe, p=0.75, list=FALSE)
subTraining <- training[subSamples, ]
subTesting <- training[-subSamples, ]
```

Exploratory Analysis

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

```
plot(as.factor(subTraining$classe), col="blue", main="Levels of the variable classe", xlab="classe level
```



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

Prediction Models

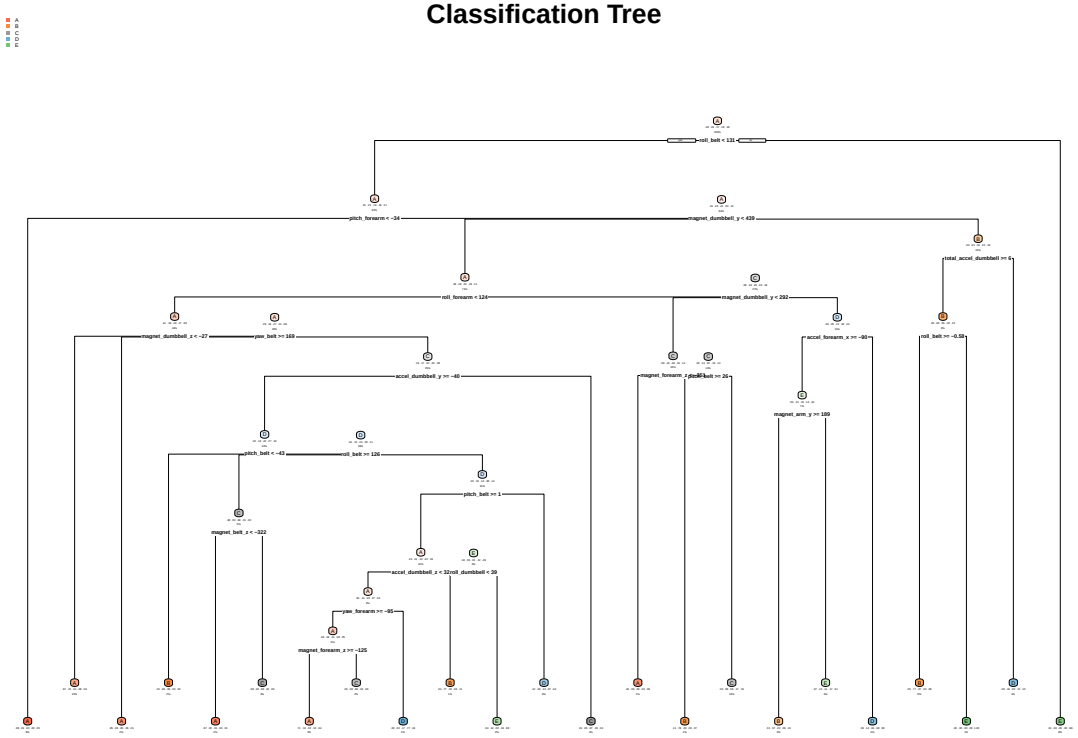
In this section a decision tree and random forest will be applied to the data.

Decision Tree

```
# Fiting the model
modFitDT <- rpart(classe ~ ., data=subTraining, method="class")
# Performing prediction
predictDT <- predict(modFitDT, subTesting, type = "class")

rpart.plot(modFitDT, main="Classification Tree",  under=TRUE, faclen=0)
```

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




```
#The following confusion matrix shows the errors of the decision tree prediction.
confusionMatrix(predictDT, as.factor(subTesting$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

```
# Fiting the model
modFitRF <- randomForest(as.factor(classe)~., data=subTraining, method="class")

# Perform prediction
predictRF <- predict(modFitRF, subTesting, type = "class")

#The following confusion matrix shows the errors of random forest prediction.
confusionMatrix(predictRF,  as.factor(subTesting$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 performs 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 chosen.

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 mis-classified.