Practical Machine Learning- Assignment

LM

2025-01-21

Background

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. 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, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

Data

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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har

Assignment

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should 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.

Loading and cleaning Data

```
library(ggplot2)
library(lattice)
library(caret)
library(kernlab)
```

```
##
## Attaching package: 'kernlab'

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

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

```r
library(knitr)
```

```r
#Load the data
trainingA <- read.csv("pml-training.csv")
testingA <- read.csv("pml-testing.csv")

#Explore the data
dim(testingA)
dim(trainingA)
head(trainingA, 5)
```

Missing values and variables with near zero variance should be removed as well as columns that are not predictors (1:7).

```r
#Delete missing values
trainingA <- read.csv("pml-training.csv", na.strings = c("NA", "#DIV/0", ""))
testingA <- read.csv("pml-testing.csv", na.strings = c("NA", "#DIV/0", ""))

#Delete columns with near zero variance
trainingA<-trainingA[,colSums(is.na(trainingA)) == 0]
testingA <-testingA[,colSums(is.na(testingA)) == 0]

#Remove first 7 columns that are not predictors
trainingA <-trainingA[,-c(1:7)]
testingA <-testingA[,-c(1:7)]

dim(trainingA)
```

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

```r
dim(testingA)
```

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

Create a data partition using the "trainingA" dataset.

Now, the data will be divided 75% in the training group and 25% in the testing group.

```r
set.seed(1000)

Part <- createDataPartition(trainingA$classe, p=0.75, list=FALSE)
training <- trainingA[Part, ]
testing <- trainingA[-Part, ]

dim(training)
```

```
## [1] 14718    53
```

```
dim(testing)
```

```
## [1] 4904   53
```

```
training$classe <- as.factor(training$classe)
testing$classe <- as.factor(testing$classe)
```

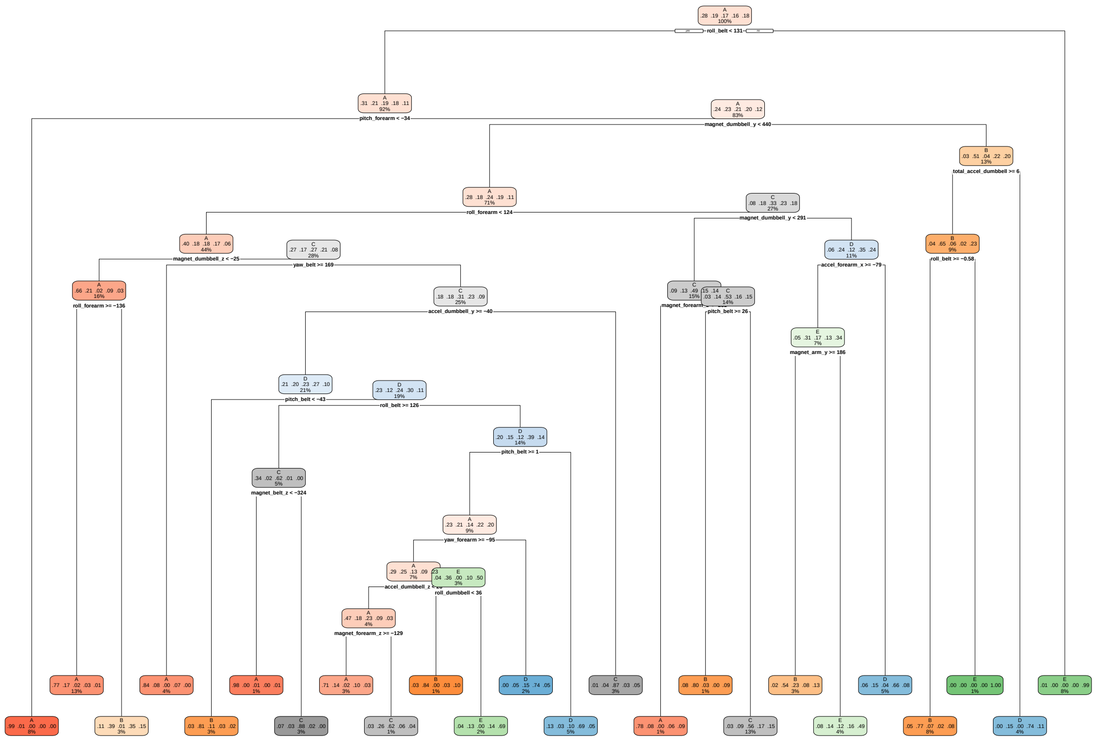
Prediction Models

Two prediction models, decision tree and random forest will be evaluated. The best model will be selected based on the value of accuracy.

Decision Tree

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

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



```
prediction_Tree <- predict(Tree, testing, type="class")
confusionMatrix(prediction_Tree, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1232  142    8   39   17
##          B   44  598   74   59   72
##          C   48   93  694  122  111
##          D   44   73   63  525   51
##          E   27   43   16   59  650
```

```
##
## Overall Statistics
##
##                Accuracy : 0.7543
##                  95% CI : (0.742, 0.7663)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.689
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8832   0.6301   0.8117   0.6530   0.7214
## Specificity            0.9413   0.9370   0.9076   0.9437   0.9638
## Pos Pred Value         0.8567   0.7060   0.6498   0.6944   0.8176
## Neg Pred Value         0.9530   0.9135   0.9580   0.9327   0.9389
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2512   0.1219   0.1415   0.1071   0.1325
## Detection Prevalence   0.2932   0.1727   0.2178   0.1542   0.1621
## Balanced Accuracy      0.9122   0.7836   0.8597   0.7983   0.8426
```

Random Forest

```
RF <- randomForest(classe~., data=training)
RF
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.48%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4183    2    0    0    0 0.0004778973
## B   12 2832    4    0    0 0.0056179775
## C    0   18 2545    4    0 0.0085703155
## D    0    0   22 2389    1 0.0095356551
## E    0    0    1    6 2699 0.0025868441
```

```
predict_RF <- predict(RF, testing)
confusionMatrix(predict_RF, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    5    0    0    0
##          B    0  939    7    0    0
##          C    0    5  846   10    0
##          D    0    0    2  793    6
##          E    0    0    0    1  895
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9927          
##                  95% CI : (0.9899, 0.9949)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9907          
##                                           
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9895   0.9895   0.9863   0.9933
## Specificity            0.9986   0.9982   0.9963   0.9980   0.9998
## Pos Pred Value         0.9964   0.9926   0.9826   0.9900   0.9989
## Neg Pred Value         1.0000   0.9975   0.9978   0.9973   0.9985
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1915   0.1725   0.1617   0.1825
## Detection Prevalence   0.2855   0.1929   0.1756   0.1633   0.1827
## Balanced Accuracy      0.9993   0.9938   0.9929   0.9922   0.9965
```

Conclusion

The Estimated Accuracy of the decision tree is 0.7484 (74.84%) and the Estimated Out-of-Sample Error is calculated as 1-accuracy and therefore results in 0.2516 (25.16%) . The Estimated Accuracy of the Random Forest Model is 0.9957 (99.57%) and the Estimated Out-of-Sample Error is 0.0046 (0.46%).Therefore, the Random Forest is the best model.

Prediction of the original testing dataset, "testingA"

```
prediction_test <- predict(RF, testingA)
prediction_test
```

```
##  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
```