Practical Machine Learning Course Project

by Przemek

Introduction

The goal of the project is to predict the manner in which the subject of the study did the exercise. The study is decribed here: http://groupware.les.inf.puc-rio.br/har and the data also comes from the study.

In the project, 2 classification models was created and the models are comparred in terms of predicting accuracy. The main meassure for the model performance is prediction accuracy on the testing model.

Data preparation

Installing the packages needed in the analysis.

```
install.packages("caret")
```

```
## Error in install.packages : Updating loaded packages
```

```
library(caret)
install.packages('e1071')
```

```
## Error in install.packages : Updating loaded packages
```

```
library(e1071)
```

Downloading the data

```
url_train='https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
url_test='https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'

train_set<-read.csv(url_train)
test_set<-read.csv(url_test)
```

As many variables in the datasets have NA or empty values, they will be removed. Also First 7 variables are useless for the classification exercise.

Removing variables with almost only NA or empty values

```
nas<-c()
empt<-c()
for(i in names(train_set)){
  nas[i]<-sum(is.na(train_set[,i]))
  empt[i]<-sum(train_set[,i]=="")
}
training<-train_set[,nas<19000&empt<19000]
testing<-test_set[,nas<19000&empt<19000]
```

Removing first 7 variables that are also useless for the classification

```
training<-training[,8:60]
testing<-testing[,8:60]
```

Spliting the training data set into training and testing subsets.The proportions 60:40.

```
set.seed(20171230)
inTrain = createDataPartition(y=training$classe, p = .60)[[1]]
trainingsubset = training[ inTrain,]
testingsubset = training[-inTrain,]
```

Classification models

Two classification models will be created:Classification Tree and Random Forests. For each model, classification matrix for testing data subset will be calculated to obtain the classification accuracy of the models.

1. Classification Tree

```
FitTREE <- train(classe~.,data=trainingsubset,method = "rpart")
print(FitTREE$finalModel)
```

```
## n= 11776
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 11776 8428 A (0.28 0.19 0.17 0.16 0.18)
##    2) roll_belt< 129.5 10697 7396 A (0.31 0.21 0.19 0.18 0.11)
##      4) pitch_forearm< -34.35 934    3 A (1 0.0032 0 0 0) *
##      5) pitch_forearm>=-34.35 9763 7393 A (0.24 0.23 0.21 0.2 0.12)
##       10) magnet_dumbbell_y< 426.5 8112 5814 A (0.28 0.18 0.24 0.19 0.1)
##         20) roll_forearm< 121.5 5048 2995 A (0.41 0.18 0.19 0.16 0.056) *
##         21) roll_forearm>=121.5 3064 2045 C (0.08 0.18 0.33 0.23 0.18) *
##       11) magnet_dumbbell_y>=426.5 1651  838 B (0.044 0.49 0.047 0.23 0.18) *
##    3) roll_belt>=129.5 1079   47 E (0.044 0 0 0 0.96) *
```

```
#plot(FitTREE$finalModel,uniform=TRUE, main="Classification FitTREE")
#text(FitTREE$finalModel,use.n=TRUE,all=TRUE,cex=.8)
TREEpredict<-predict(FitTREE,testingsubset)
confusionMatrix(testingsubset$classe,TREEpredict)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1990   51  164    0   27
##          B  579  535  404    0    0
##          C  609   51  708    0    0
##          D  570  230  486    0    0
##          E  199  191  385    0  667
##
## Overall Statistics
##
##                Accuracy : 0.4971
##                  95% CI : (0.4859, 0.5082)
##     No Information Rate : 0.5031
##     P-Value [Acc > NIR] : 0.8583
##
##                   Kappa : 0.3441
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.5042  0.50567  0.32976       NA  0.96110
## Specificity            0.9379  0.85519  0.88419   0.8361  0.89164
## Pos Pred Value         0.8916  0.35244  0.51754       NA  0.46255
## Neg Pred Value         0.6514  0.91735  0.77786       NA  0.99578
## Prevalence             0.5031  0.13485  0.27364   0.0000  0.08845
## Detection Rate         0.2536  0.06819  0.09024   0.0000  0.08501
## Detection Prevalence   0.2845  0.19347  0.17436   0.1639  0.18379
## Balanced Accuracy      0.7211  0.68043  0.60698       NA  0.92637
```

The out of sample accuracy is 49,71% which is not a good result.

1. Random Forest Model

```
FitRF5 <- train(classe~.,data=trainingsubset,method = "rf",trControl=trainControl(method = "cv", number = 5))
print(FitRF5$finalModel)
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.79%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3343    4    0    0    1 0.001493429
## B   17 2252    9    0    1 0.011847301
## C    0   13 2031   10    0 0.011197663
## D    0    1   23 1903    3 0.013989637
## E    0    2    2    7 2154 0.005080831
```

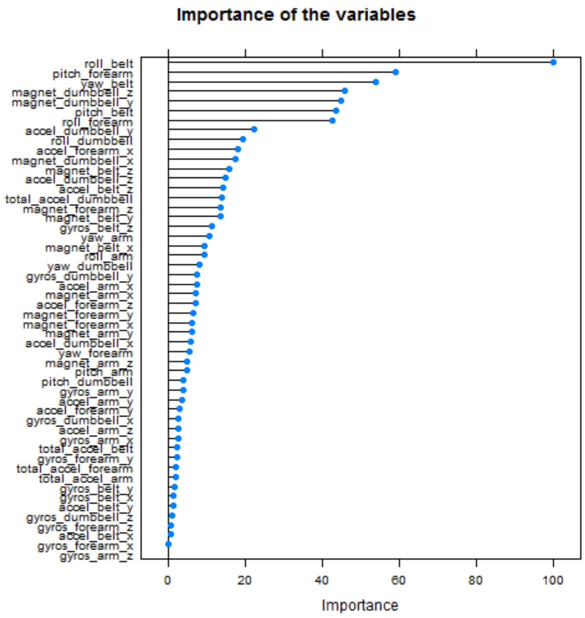
```
RF5predict<-predict(FitRF5,testingsubset)
confusionMatrix(testingsubset$classe,RF5predict)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    2    1    0    0
##          B   19 1493    6    0    0
##          C    0    8 1358    2    0
##          D    0    2   21 1263    0
##          E    0    1    4    5 1432
## 
## Overall Statistics
##                                           
##                Accuracy : 0.991           
##                  95% CI : (0.9886, 0.9929)
##     No Information Rate : 0.2865          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9886          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9915   0.9914   0.9770   0.9945   1.0000
## Specificity            0.9995   0.9961   0.9985   0.9965   0.9984
## Pos Pred Value         0.9987   0.9835   0.9927   0.9821   0.9931
## Neg Pred Value         0.9966   0.9979   0.9951   0.9989   1.0000
## Prevalence             0.2865   0.1919   0.1772   0.1619   0.1825
## Detection Rate         0.2841   0.1903   0.1731   0.1610   0.1825
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9955   0.9937   0.9877   0.9955   0.9992
```

Random Forests Classification seems to be the most accurate. 99,06% of the observations in the testing subset was classified correctly.

To explore the model, variable importance plot was created

```
plot(varImp(FitRF5),main="Importance of the variables")
```



Predicting the values for the test set.

```
test_set$PredictedClasse<-predict(FitRF5,testing)
test_set$PredictedClasse
```

```
##  [1] 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
```

Summary

Random Forrest classification gives a very good results for the classification. In the out of sample classification, the accuracy war 99%. It was much higher comparred to the Classification Tree. Therefore, the Random Forest Classification was chosen as a final model.