

Machine Learning Project Assignment

Weight lifting exercise - Predict if exercises are done correctly

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 dumbbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here:

<http://groupware.les.inf.puc-rio.br/har> (<http://groupware.les.inf.puc-rio.br/har>) (see the section on the Weight Lifting Exercise Dataset).

The training data for this project are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv>
(<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv>)

The test data are available here:

<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv>
(<https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv>)

Reference Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidui, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6. <http://groupware.les.inf.puc-rio.br/work.jsf?p1=10335> (<http://groupware.les.inf.puc-rio.br/work.jsf?p1=10335>)

Objective Develop a prediction model using the training data set to predict Classe.

Loading & Exploring Data

```
# Loading libraries & enable parallel processing
library(caret);library(gbm);library(randomForest);library(doParallel)
set.seed(12345)
cl <- makeCluster(detectCores());registerDoParallel(cl)
```

```
training.rd <- read.csv("C:/Users/Werner/Documents/rcourse/pml01/pmldata/pml-training.csv")
testing.rd <- read.csv("C:/Users/Werner/Documents/rcourse/pml01/pmldata/pml-testing.csv")
```

```
dim(training.rd);dim(testing.rd);
```

```
## [1] 19622 160
```

```
## [1] 20 160
```

With 19622 rows and 160 columns the training data set is huge. Using commands like `View(training.rd)` and `edit(training.rd)` the need for cleaning data is obvious (empty columns, NA columns). Also columns 1 to 7 are not related to measurements. Not related columns and NA columns are eliminated. In case I have trouble to develop a very predictive model I would come back to the data cleaning activities. As the number of columns is huge I also use `nearZeroVar(t)` to eliminate columns with small variances.

```
t<- training.rd[,!sapply(training.rd,function(x) any(is.na(x))))]
t <- t[,-c(1:7)]
nvar <- nearZeroVar(t)
tm <- t[,-nvar]; dim(tm)
```

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

Test & Validation Data The 19622 rows are splitted. 13737 rows are used to develop the model, 5885 rows are kept aside to validate the model(s)

```
# create training set indexes with 70% of data
inTrain <- createDataPartition(y=tm$classe,p=0.70, list=FALSE)
training.dt <- tm[inTrain,]
val.dt <- tm[-inTrain,]
dim(training.dt); dim(val.dt)
```

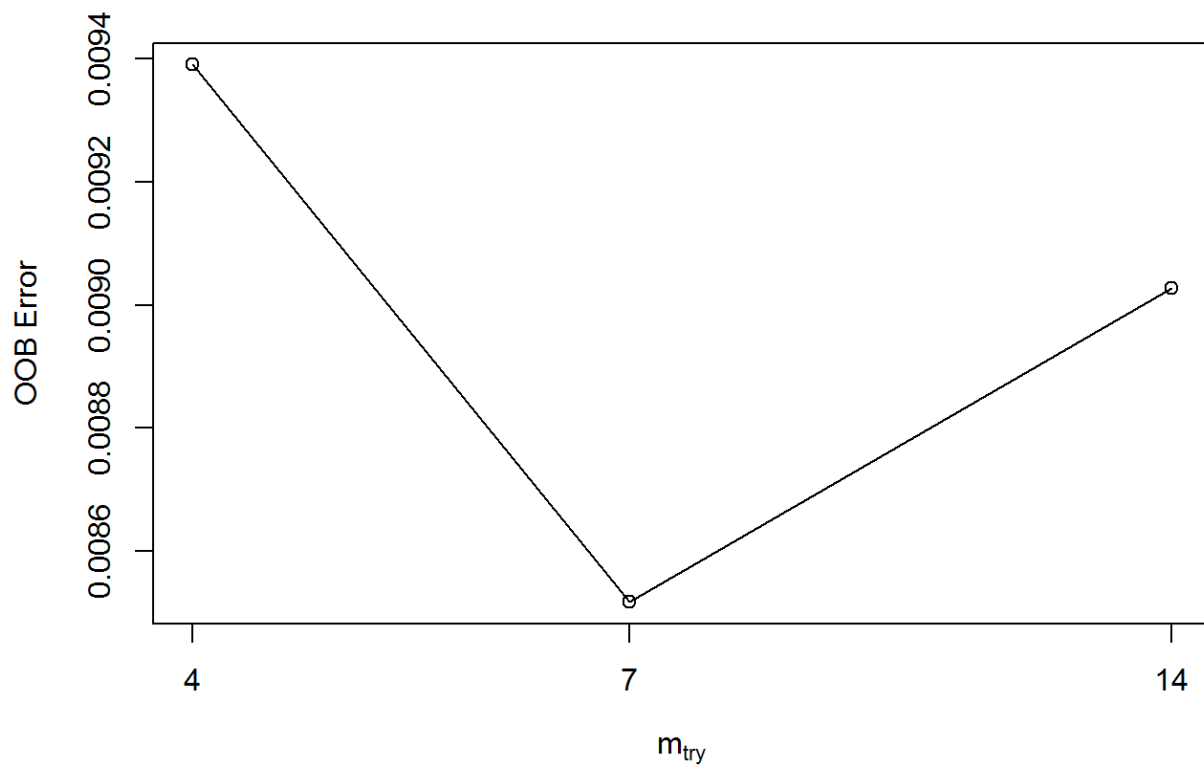
```
## [1] 13737 53
```

```
## [1] 5885 53
```

Model Development As we have a classification problem (5 classes) a very popular and accurate method is random forest tree (RF) along with boosting. The 2nd method is therefore is gradient boosting (GBM). As RF showed very accurate prediction with default values it was the model of choice. I tried to improve th default RF model but was not able to improve it with `mtry`.

```
tuneRF(training.dt[, -53], training.dt[, 53])
```

```
## mtry = 7   OOB error = 0.85%
## Searching left ...
## mtry = 4     OOB error = 0.94%
## -0.1025641 0.05
## Searching right ...
## mtry = 14    OOB error = 0.9%
## -0.05982906 0.05
```



```
##      mtry   OOBError
## 4.OOB     4 0.009390697
## 7.OOB     7 0.008517143
## 14.OOB    14 0.009026716
```

```
f1 <- randomForest(classe ~ ., data = training.dt)
f2 <- train(classe ~ ., data = training.dt, method="gbm", verbose=FALSE)
f3 <- randomForest(classe ~ ., data = training.dt, mtry=7)
```

Model Validation The 3 models are tested against the training data set and the validation training set

```
pred.t1 <- predict(f1, training.dt)
c.t1<- confusionMatrix(training.dt$classe,pred.t1)

pred.t2 <- predict(f2, training.dt)
c.t2<- confusionMatrix(training.dt$classe,pred.t2)

pred.t3 <- pred.t3 <- predict(f3, training.dt)
c.t3 <- confusionMatrix(training.dt$classe,pred.t3)

pred.v1 <- predict(f1, val.dt)
c.v1 <- confusionMatrix(val.dt$classe,pred.v1)

pred.v2 <- predict(f2, val.dt)
c.v2<- confusionMatrix(val.dt$classe,pred.v2)

pred.v3 <- predict(f3, val.dt)
c.v3 <- confusionMatrix(val.dt$classe,pred.v3)
```

Print Validation for Training Data Set

```
c.t1; c.t2;c.t3
```

```

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##           A 3906    0    0    0    0
##           B    0 2658    0    0    0
##           C    0    0 2396    0    0
##           D    0    0    0 2252    0
##           E    0    0    0    0 2525
##
## Overall Statistics
##
##           Accuracy : 1
##           95% CI : (0.9997, 1)
## No Information Rate : 0.2843
## P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 1
## McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity          1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity          1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value       1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value       1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence           0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate       0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence 0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy    1.0000   1.0000   1.0000   1.0000   1.0000

```

```

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A      B      C      D      E
##           A 3866    25    11     3     1
##           B   57 2553    47     0     1
##           C    0   51 2322    19     4
##           D    2    2   69 2173     6
##           E    2   10   14   28 2471
##
## Overall Statistics
##
##           Accuracy : 0.9744
##           95% CI : (0.9716, 0.977)
##           No Information Rate : 0.2859
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.9676
##           McNemar's Test P-Value : 2.279e-13
##
## Statistics by Class:
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity          0.9845   0.9667   0.9428   0.9775   0.9952
## Specificity          0.9959   0.9905   0.9934   0.9931   0.9952
## Pos Pred Value       0.9898   0.9605   0.9691   0.9649   0.9786
## Neg Pred Value       0.9938   0.9921   0.9876   0.9956   0.9989
## Prevalence           0.2859   0.1923   0.1793   0.1618   0.1808
## Detection Rate       0.2814   0.1858   0.1690   0.1582   0.1799
## Detection Prevalence 0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy     0.9902   0.9786   0.9681   0.9853   0.9952

```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##           A 3906    0    0    0    0
##           B    0 2658    0    0    0
##           C    0    0 2396    0    0
##           D    0    0    0 2252    0
##           E    0    0    0    0 2525
##
## Overall Statistics
##
##           Accuracy : 1
##           95% CI : (0.9997, 1)
## No Information Rate : 0.2843
## P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 1
## McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity           1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity           1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value        1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value        1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence            0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate        0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence  0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

Print Validation for Validation Data Set

```
c.v1; c.v2;c.v3
```

```

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A      B      C      D      E
##           A 1673      1      0      0      0
##           B   6 1130      3      0      0
##           C   0      7 1016      3      0
##           D   0      0   11  952      1
##           E   0      0      0      3 1079
##
## Overall Statistics
##
##           Accuracy : 0.9941
##           95% CI : (0.9917, 0.9959)
##           No Information Rate : 0.2853
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.9925
##           McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity           0.9964   0.9930   0.9864   0.9937   0.9991
## Specificity           0.9998   0.9981   0.9979   0.9976   0.9994
## Pos Pred Value         0.9994   0.9921   0.9903   0.9876   0.9972
## Neg Pred Value         0.9986   0.9983   0.9971   0.9988   0.9998
## Prevalence             0.2853   0.1934   0.1750   0.1628   0.1835
## Detection Rate         0.2843   0.1920   0.1726   0.1618   0.1833
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy       0.9981   0.9955   0.9922   0.9957   0.9992

```



```

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A     B     C     D     E
##           A 1643    24     7     0     0
##           B   26 1087    23     1     2
##           C    0   34  983     9     0
##           D    3    3   31  925     2
##           E    3   13   11   15 1040
##
## Overall Statistics
##
##           Accuracy : 0.9648
##           95% CI : (0.9598, 0.9694)
##           No Information Rate : 0.2846
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.9555
##           McNemar's Test P-Value : 1.165e-08
##
## Statistics by Class:
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity           0.9809    0.9363    0.9318    0.9737    0.9962
## Specificity           0.9926    0.9890    0.9911    0.9921    0.9913
## Pos Pred Value        0.9815    0.9543    0.9581    0.9595    0.9612
## Neg Pred Value        0.9924    0.9844    0.9852    0.9949    0.9992
## Prevalence            0.2846    0.1973    0.1793    0.1614    0.1774
## Detection Rate        0.2792    0.1847    0.1670    0.1572    0.1767
## Detection Prevalence  0.2845    0.1935    0.1743    0.1638    0.1839
## Balanced Accuracy      0.9868    0.9626    0.9614    0.9829    0.9937

```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A     B     C     D     E
##           A 1673     1     0     0     0
##           B    6 1130     3     0     0
##           C    0    10 1013     3     0
##           D    0     0    11  952     1
##           E    0     0     0    3 1079
##
## Overall Statistics
##
##           Accuracy : 0.9935
##           95% CI : (0.9911, 0.9954)
##           No Information Rate : 0.2853
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.9918
##           McNemar's Test P-Value : NA
##
## Statistics by Class:
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity           0.9964   0.9904   0.9864   0.9937   0.9991
## Specificity           0.9998   0.9981   0.9973   0.9976   0.9994
## Pos Pred Value        0.9994   0.9921   0.9873   0.9876   0.9972
## Neg Pred Value        0.9986   0.9977   0.9971   0.9988   0.9998
## Prevalence            0.2853   0.1939   0.1745   0.1628   0.1835
## Detection Rate        0.2843   0.1920   0.1721   0.1618   0.1833
## Detection Prevalence  0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9981   0.9942   0.9918   0.9957   0.9992
```

In/Out Error Rates in-sample errors t1 had no in-errors, t2 had about 2.5% in-errors and t3 again had no in-errors out of sample errors Doing the prediction against t1, t2, t3 showed an accuracy of 99.41%, 96.48% and 99.34%

Prediction Model f1 (the model with the lowest error rate related to the validation set is used to predict the classifications for the test data set (20 observation) The model was able to predict the outcome correctly(no errors)

```
pred.r <- predict(f1, testing.rd)
pred.r
```

```
##  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
```

```
# code for submitting the predictions commented out
#answers = rep("X", 20)
# pml_write_files = function(x){
#   n = length(x)
#   for(i in 1:n){
#     filename = paste0("problem_id_",i,".txt")
#     write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
#   }
# }

# pml_write_files(pred.r)
```