Practicle Machine Learning Assingment

2023-04-20

SYNOPOSIS

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. I will create a report describing how I have built my model, how I have used cross validation, what I think the expected out of sample error is, and why I have made the choices I did. I will also use your prediction model to predict 20 different test cases.

Data Sources

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)

The data for this project come from this source:

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

Loading The Data and Libraries

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.3
```

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

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

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

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

```
## Warning: package 'rpart.plot' was built under R version 4.2.3
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.2.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

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

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.2.3
```

```
## Loaded gbm 2.1.8.1
```

```
training_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testing_url <-  "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

training <- read.csv(url(training_url))
testing <- read.csv(url(testing_url))

dim(training)
```

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

```
dim(testing)
```

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

Data cleaning

there are 3 parts in data cleaning

A. Removing values which are having nearly Zero Variance .

```
nearZeroVar <- nearZeroVar(training)
training <- training[, - nearZeroVar]
teasting <- testing[ , - nearZeroVar]
dim(training)
```

```
## [1] 19622   100
```

```
dim(testing)
```

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

removing variables which are mostly NA

```
allna <- sapply(training, function(x)mean(is.na(x))) > 0.95
training <- training[, allna==FALSE]
testing <- testing[,allna==FALSE]
dim(training)
```

```
## [1] 19622    59
```

```
dim(testing)
```

```
## [1] 20 94
```

Subset data

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

Data Partitioning / Cross validation

In this section cross-validation will be performed by splitting the training data in training 60% and testing 40% data.

```
intrain <- createDataPartition(y = training$classe , p= .6 , list = FALSE )
subtraining <- training[intrain , ]
subtesting <- training[ -intrain , ]
dim(subtraining)
```

```
## [1] 11776    52
```

```
dim(subtesting)
```

```
## [1] 7846   52
```

Random forest Model

prediction in term of Random forest Model

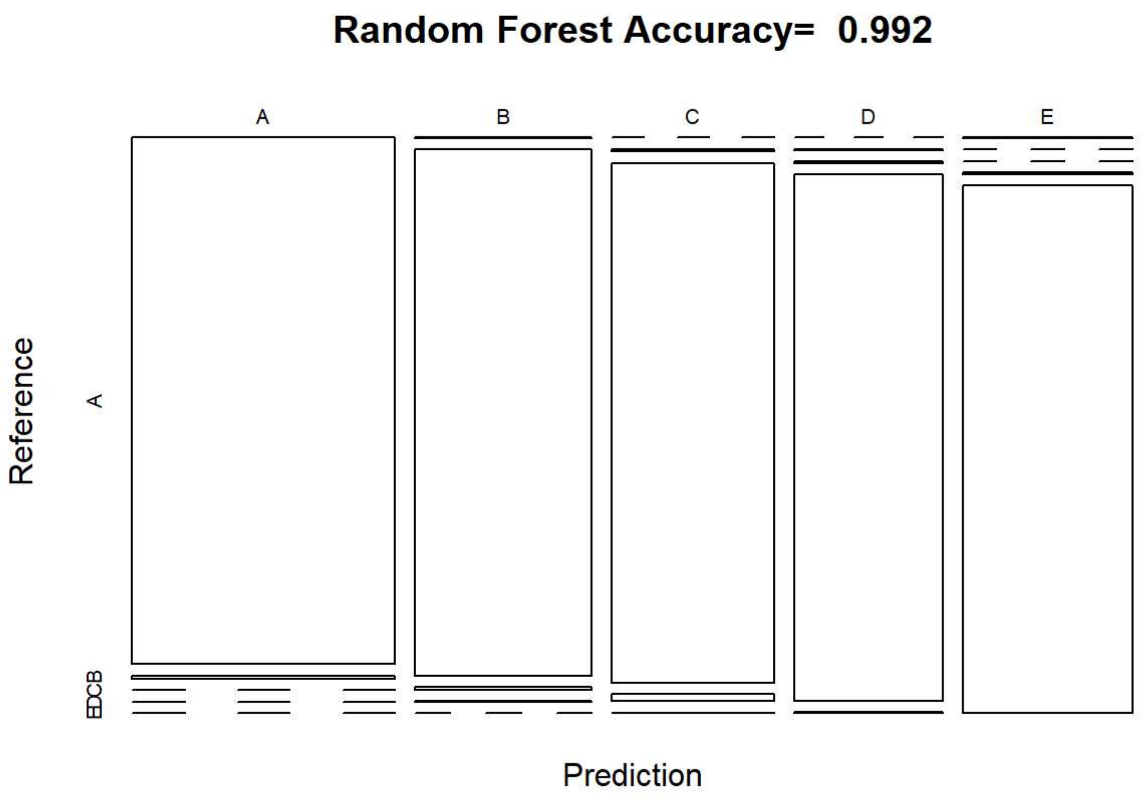
```
set.seed(111)
controlRF <- trainControl(method="cv", number=3, verboseIter=FALSE)
RF_modelfit <- train(classe ~ . , data = subtraining , method = "rf" , trControl = controlRF
, ntree = 100)
RF_modelfit$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, ntree = 100, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 26
##
##         OOB estimate of  error rate: 0.91%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3341    6    1    0    0 0.002090800
## B   20 2242   12    3    2 0.016235191
## C    0   12 2032   10    0 0.010710808
## D    0    1   26 1901    2 0.015025907
## E    0    2    5    5 2153 0.005542725
```

```
RF_predict <- predict(RF_modelfit , newdata = subtesting)
CF_randomforest <- confusionMatrix(RF_predict,  as.factor( subtesting$classe))
CF_randomforest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2230   12    0    0    0
##          B    1 1497    8    1    0
##          C    0    7 1356   19    1
##          D    0    2    4 1261    2
##          E    1    0    0    5 1439
##
## Overall Statistics
##
##                Accuracy : 0.992
##                  95% CI : (0.9897, 0.9938)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9898
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9862   0.9912   0.9806   0.9979
## Specificity            0.9979   0.9984   0.9958   0.9988   0.9991
## Pos Pred Value         0.9946   0.9934   0.9805   0.9937   0.9958
## Neg Pred Value         0.9996   0.9967   0.9981   0.9962   0.9995
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1908   0.1728   0.1607   0.1834
## Detection Prevalence   0.2858   0.1921   0.1763   0.1617   0.1842
## Balanced Accuracy      0.9985   0.9923   0.9935   0.9897   0.9985
```

```
plot(CF_randomforest$table ,col= CF_randomforest$byClass ,  main = paste("Random Forest Accur
acy= " , round(CF_randomforest$overall['Accuracy'], 4  )))
```



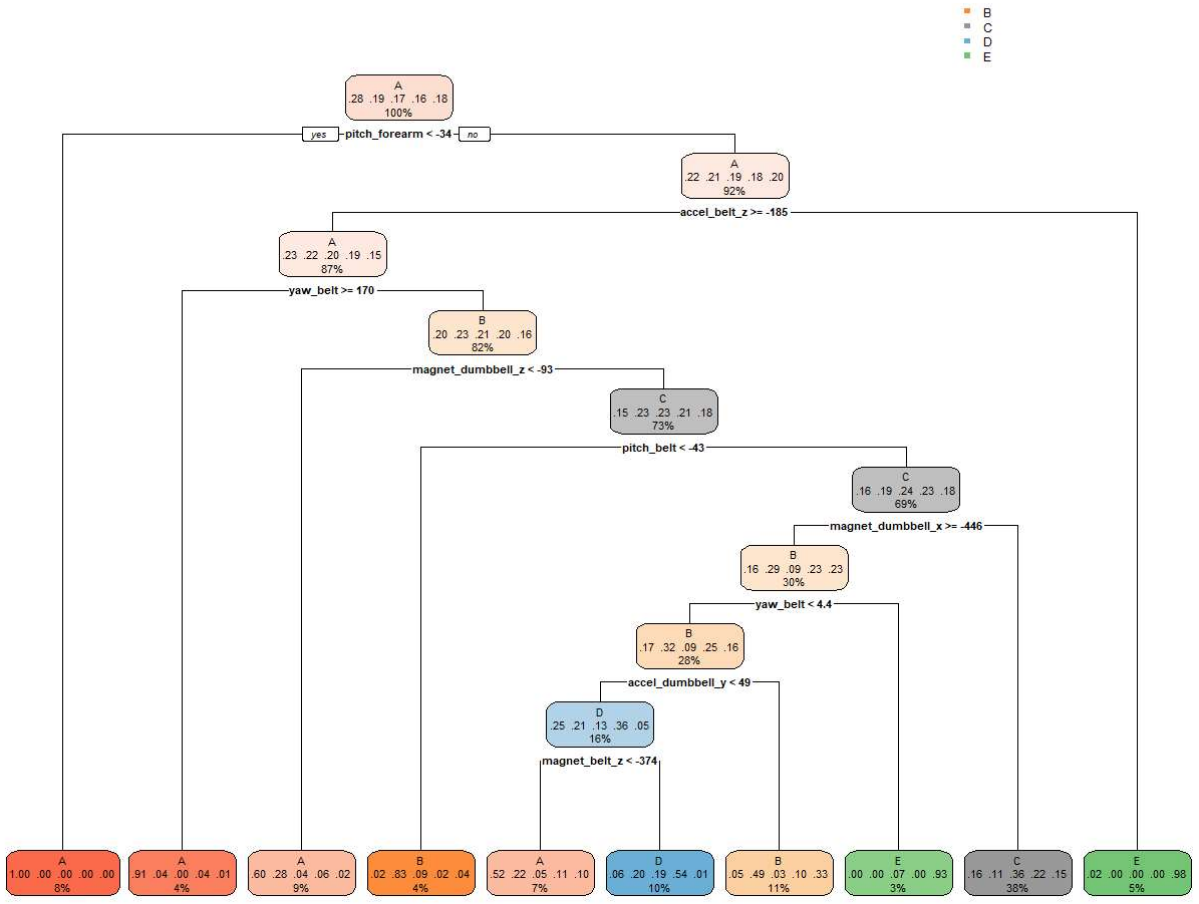
Decision Tree Model

prediction in term of Decision tree model

```
modfit <- train(classe ~ ., data= subtraining , method = "rpart")
prediction <- predict(modfit , newdata = subtesting)
confusionMatrix(prediction , as.factor( subtesting$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1656  345   75  126   69
##          B   48  695   55   89  292
##          C  474  365 1075  700  446
##          D   36  112  152  371    5
##          E   18    1   11    0  630
##
## Overall Statistics
##
##                Accuracy : 0.5642
##                  95% CI : (0.5532, 0.5752)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4491
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7419  0.45784   0.7858  0.28849  0.43689
## Specificity            0.8905  0.92351   0.6936  0.95351  0.99532
## Pos Pred Value         0.7292  0.58948   0.3513  0.54882  0.95455
## Neg Pred Value         0.8967  0.87656   0.9388  0.87238  0.88700
## Prevalence             0.2845  0.19347   0.1744  0.16391  0.18379
## Detection Rate         0.2111  0.08858   0.1370  0.04729  0.08030
## Detection Prevalence   0.2894  0.15027   0.3900  0.08616  0.08412
## Balanced Accuracy      0.8162  0.69068   0.7397  0.62100  0.71610
```

```
rpart.plot(modfit$finalModel , roundint = F)
```


B
C
D
E
A .28 .19 .17 .16 .18 100%
yes
pitch_forearm < -34
no
A .22 .21 .19 .18 .20 92%
accel_belt_z >= -185
A .23 .22 .20 .19 .15 87%
yaw_belt >= 170
B .20 .23 .21 .20 .16 82%
magnet_dumbbell_z < -93
C .15 .23 .23 .21 .18 73%
pitch_belt < -43
C .16 .19 .24 .23 .18 69%
magnet_dumbbell_x >= -446
B .16 .29 .09 .23 .23 30%
yaw_belt < 4.4
B .17 .32 .09 .25 .16 28%
accel_dumbbell_y < 49
D .25 .21 .13 .36 .05 16%
magnet_belt_z < -374
A 1.00 .00 .00 .00 .00 8%
A .91 .04 .00 .04 .01 4%
A .60 .28 .04 .06 .02 9%
B .02 .83 .09 .02 .04 4%
A .52 .22 .05 .11 .10 7%
D .06 .20 .19 .54 .01 10%
B .05 .49 .03 .10 .33 11%
E .00 .00 .07 .00 .93 3%
C .16 .11 .36 .22 .15 38%
E .02 .00 .00 .00 .98 5%

Generalised Boosted Method

prediction in term of General boosted Method

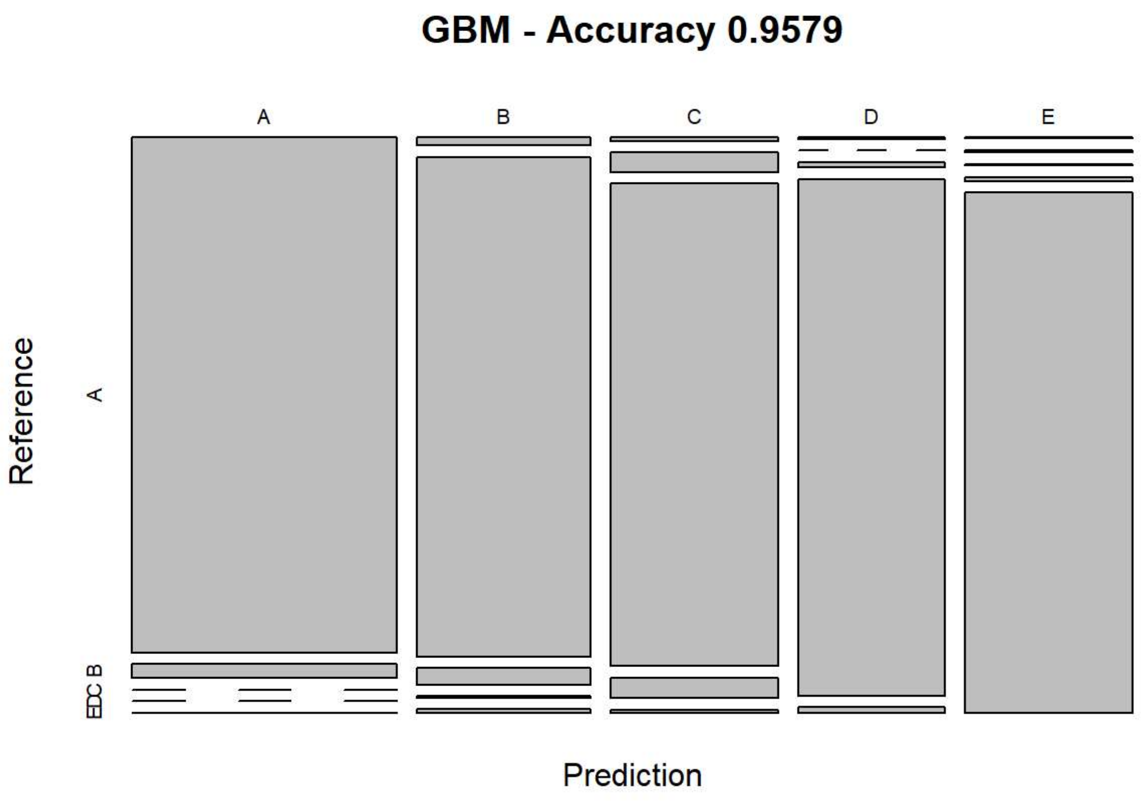
```
set.seed(112)
controlGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
gbm_modfit <- train(classe ~ . , data= subtraining , method = "gbm" ,trControl = controlGBM,
verbose = F)
gbm_modfit$finalmodel
```

```
## NULL
```

```
gbm_prediction <- predict(gbm_modfit , newdata = subtesting)
conf_gbm_prediction <- confusionMatrix(gbm_prediction ,as.factor( subtesting$classe))
conf_gbm_prediction
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 2195   60    0    0    3
##          B   22 1397   46    5   11
##          C   10   53 1305   56    9
##          D    3    0   14 1214   14
##          E    2    8    3   11 1405
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9579          
##                  95% CI : (0.9533, 0.9623)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9468          
##                                           
##  Mcnemar's Test P-Value : 3.492e-10       
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9834   0.9203   0.9539   0.9440   0.9743
## Specificity            0.9888   0.9867   0.9802   0.9953   0.9963
## Pos Pred Value         0.9721   0.9433   0.9107   0.9751   0.9832
## Neg Pred Value         0.9934   0.9810   0.9902   0.9891   0.9942
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2798   0.1781   0.1663   0.1547   0.1791
## Detection Prevalence   0.2878   0.1888   0.1826   0.1587   0.1821
## Balanced Accuracy      0.9861   0.9535   0.9671   0.9696   0.9853
```

```
plot(conf_gbm_prediction$table , col = conf_gbm_prediction$classe , main= paste("GBM - Accura
cy" , round(conf_gbm_prediction$overall['Accuracy'],4)))
```



Applying selected model on the Test data

The accuracy of the 2 regression modeling methods above are: Random Forest : 0.9993 GBM : 0.9874 In that case, the Random Forest model will be applied to predict the quiz.

Conclusion

The overall performance of rf is superior compared to the other two models, rpart and glmnet, on the training and validation dataset. Overall the accuracy of rpart and glmnet is lower than I expected. Per□formance of the glmnet could be further improved by choosing higher lambda-values. Lower lambda-values provoke a warning message that no convergence can be found after the maximum amount of iterations. Both models could improve by choosing a better parameter set. The performance of all models is is a bit lower on the validation data, but overall near the accuracy of the training data implying a low OOS, which is less than 1 % according to the model metrics for rf (0.08%, see: Metrics for Random Forest), and rejecting an overfitting of the train data split as seen in the validation data prediction accuracy. This low OOS-error is as expected. Though I suspect, given extremly high accuracy of the rf model, that this model would perform poorly in a real world setting with other participiants and other measuring devices. A problem known for machine learning. This is further described in this paper for example. This could be avoided by removing predictors that are highly specific to this data set, like the user_name, num_window or cvtd_timestamp and aquiring a much bigger data set with more devices and more diverse users. cvtd_timestamp seems to have a reasonable high impact on the prediction but has no impact in a real world setting (see: Predictor Impact in the Annex). Given the near perfect accuracy of the rf model on the validation dataset no further modeling techniques like ensemble models or further parameter tuning is performed and this model is choosen for predicting the testing data set.