Practical Machine Learning Final Project

Renga Pandurengan

2022-12-16

Project Instructions

What needs to be submitted

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.

Peer Review Portion

Your submission for the Peer Review portion should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).

Course Project Prediction Quiz Portion

Apply your machine learning algorithm to the 20 test cases available in the test data above and submit your predictions in appropriate format to the Course Project Prediction Quiz for automated grading.

Reproducibility

Due to security concerns with the exchange of R code, your code will not be run during the evaluation by your classmates. Please be sure that if they download the repo, they will be able to view the compiled HTML version of your analysis.

Loading Required Packages:

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.2
```

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.2.2
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.2
```

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

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 4.2.2
```

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

```
## Warning: package 'rpart.plot' was built under R version 4.2.2
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 4.2.2
```

```
## Loading required package: tibble
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.2.2
```

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

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

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

```
## The following object is masked from 'package:rattle':
##
##     importance
```

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

```
library(kernlab)
```

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

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

Downloading train and test datasets

```
train = read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")
test =read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")
```

Data Clensing:

- Eliminating Variables/Columns with "NA" values
- Eliminating Variables/Columns that is not related to outcome variable
- Eliminating Variables/Columns that have less variability

```
train1 <-train[ , colSums(is.na(train))==0]#Removing Variables/Columns with "NA" values
test1 <- test[, colSums(is.na(test))==0]
train2 <- train1[,-c(1:7)]#removing first seven variables/Columns
nsv <- nearZeroVar(train2)#Removing Variables/Columns that is not related to outcome variable
train3 <- train2[,-nsv]
```

Cleaned data set to be used for modeling had 53 variables and 19622 observations

Partioning the data

```
set.seed(1234)
inTrain <- createDataPartition(y=train3$classe, p=0.7, list=F)
training <- train3[inTrain,]
validity <- train3[-inTrain,]
```

Of the modelling data set, training data set had 53 variables and 13737 observations and validation data set had 53 variables and 5885 observations.

Cross-Validation

Splitting the training set into traning/testing sets and building model on training set. The evaluation is done on testing set.The process is repeated , the average the estimated errors.The cross-validation is used

1. to pick variables to include in the model,
2. to pick the type of predication function to use,
3. to pick parameters in the prediction function, and
4. comparing different predictors. Cross validation is done for each model with K=3.

Data Modelling

Four models will be built for this project and pick the one which provides the best accuracy. * 1. Decision tree(rpart) * 2. Gradient boosting tree(GBM) * 3. Random Forest (RF) * 4. Support Vector Machine (SVM)

```
fitControl <- trainControl(method='cv', number = 5)
```

Prediction models

a) Decision Tree Model

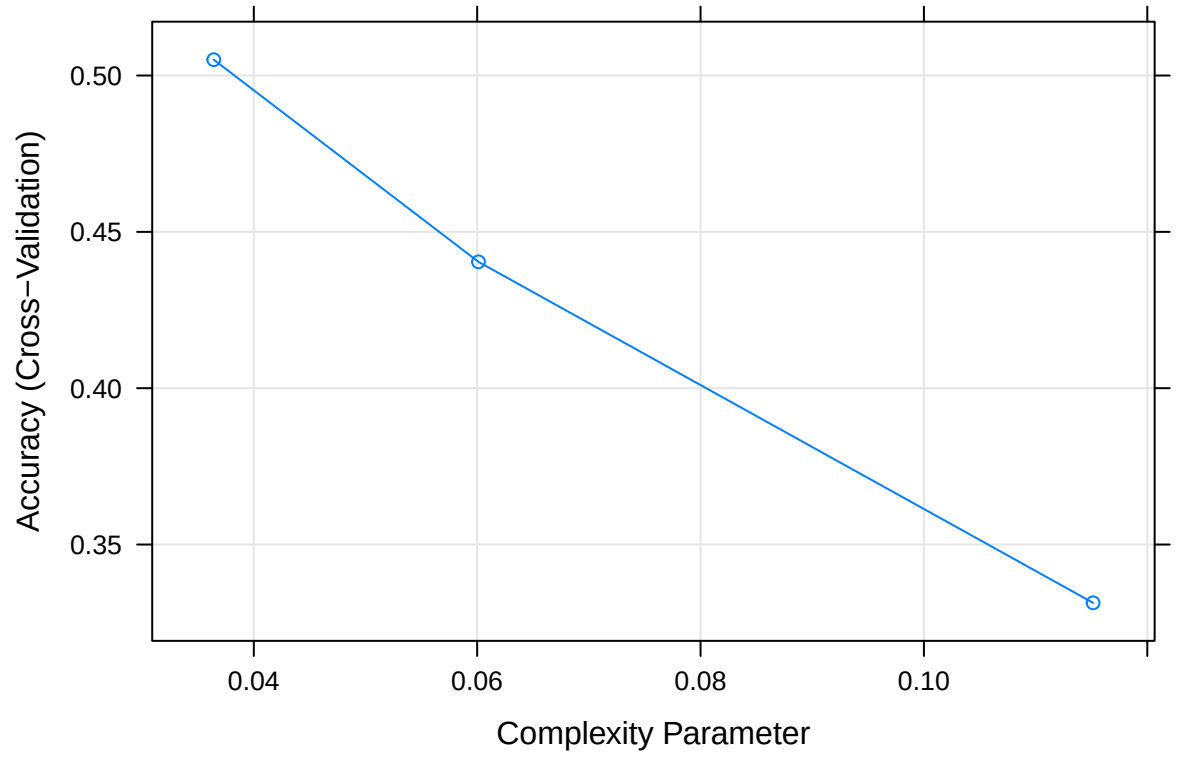
```
# Model fit
set.seed(1234)
RPART <- train(  classe ~ .,  data=training,  trControl=fitControl,  method='rpart')
save(RPART, file='./ModelFitRPART.RData')

# Prediction on validation data set
predRPART <- predict(RPART, newdata=validity)
cmRPART <-confusionMatrix(predRPART, factor(validity$classe))
cmRPART
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1519  473  484  451  156
##          B   28  401   45  167  148
##          C  123  265  497  346  289
##          D    0    0    0    0    0
##          E    4    0    0    0  489
##
## Overall Statistics
##
##                Accuracy : 0.4938
##                  95% CI : (0.4809, 0.5067)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.338
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9074  0.35206  0.48441   0.0000  0.45194
## Specificity            0.6286  0.91825  0.78946   1.0000  0.99917
## Pos Pred Value         0.4927  0.50824  0.32697      NaN  0.99189
## Neg Pred Value         0.9447  0.85518  0.87881   0.8362  0.89002
## Prevalence             0.2845  0.19354  0.17434   0.1638  0.18386
```

```
## Detection Rate         0.2581  0.06814  0.08445   0.0000  0.08309
## Detection Prevalence   0.5239  0.13407  0.25828   0.0000  0.08377
## Balanced Accuracy      0.7680  0.63516  0.63693   0.5000  0.72555
```

```
# Plotting the model
plot(RPART)
```



b) Generalized Boosted Model

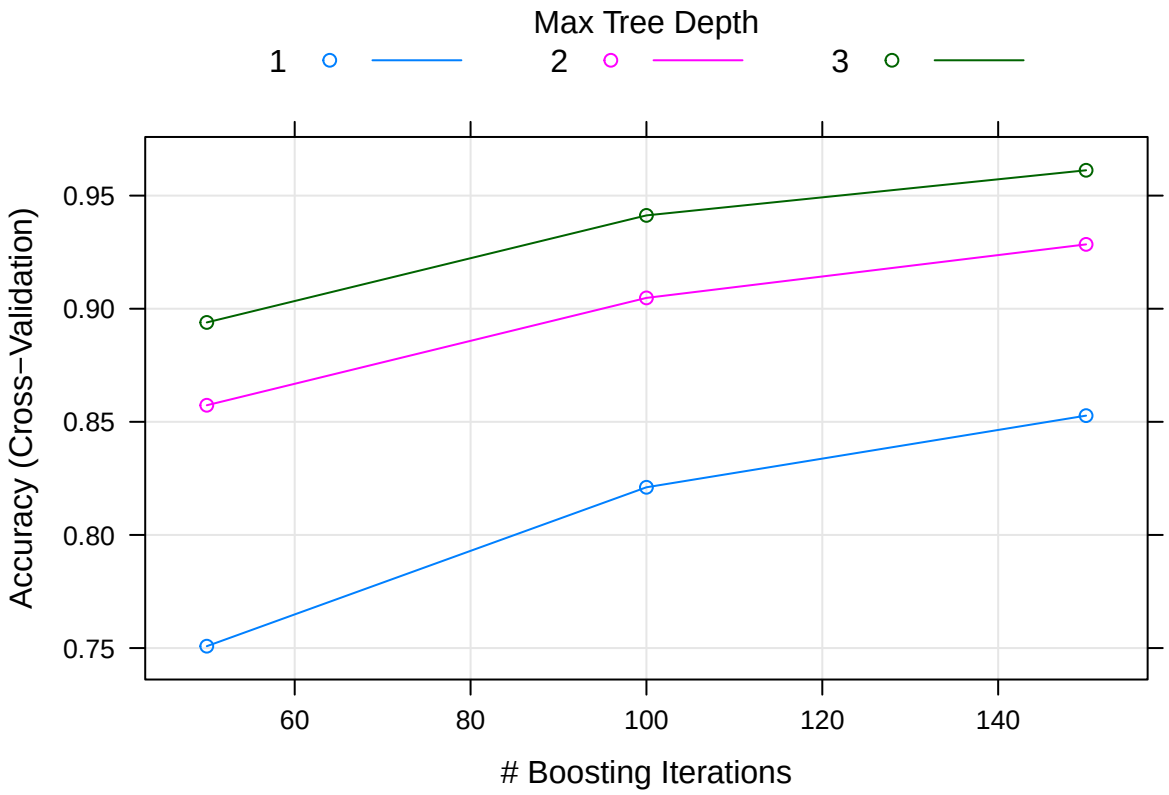
```
# Model fit
set.seed(1234)
GBM <- train(  classe ~ .,   data=training,  trControl=fitControl,  method='gbm',verbose = FALSE)
save(GBM, file='./ModelFitGBM.RData')

# Prediction on validation data set
predGBM <- predict(GBM, newdata=validity)
cmGBM <-confusionMatrix(predGBM, factor(validity$classe))
cmGBM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
```

```
##          A 1653   27    0    3    1
##          B   10 1083   36    3   12
##          C    8   29  977   26   10
##          D    2    0   12  927    8
##          E    1    0    1    5 1051
##
## Overall Statistics
##
##                Accuracy : 0.967
##                  95% CI : (0.9622, 0.9714)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9583
##
##  Mcnemar's Test P-Value : 2.194e-06
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9875   0.9508   0.9522   0.9616   0.9713
## Specificity            0.9926   0.9871   0.9850   0.9955   0.9985
## Pos Pred Value         0.9816   0.9467   0.9305   0.9768   0.9934
## Neg Pred Value         0.9950   0.9882   0.9899   0.9925   0.9936
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2809   0.1840   0.1660   0.1575   0.1786
## Detection Prevalence   0.2862   0.1944   0.1784   0.1613   0.1798
## Balanced Accuracy      0.9900   0.9690   0.9686   0.9786   0.9849
```

```
# Plotting the model
plot(GBM)
```



c) Random Forest Model

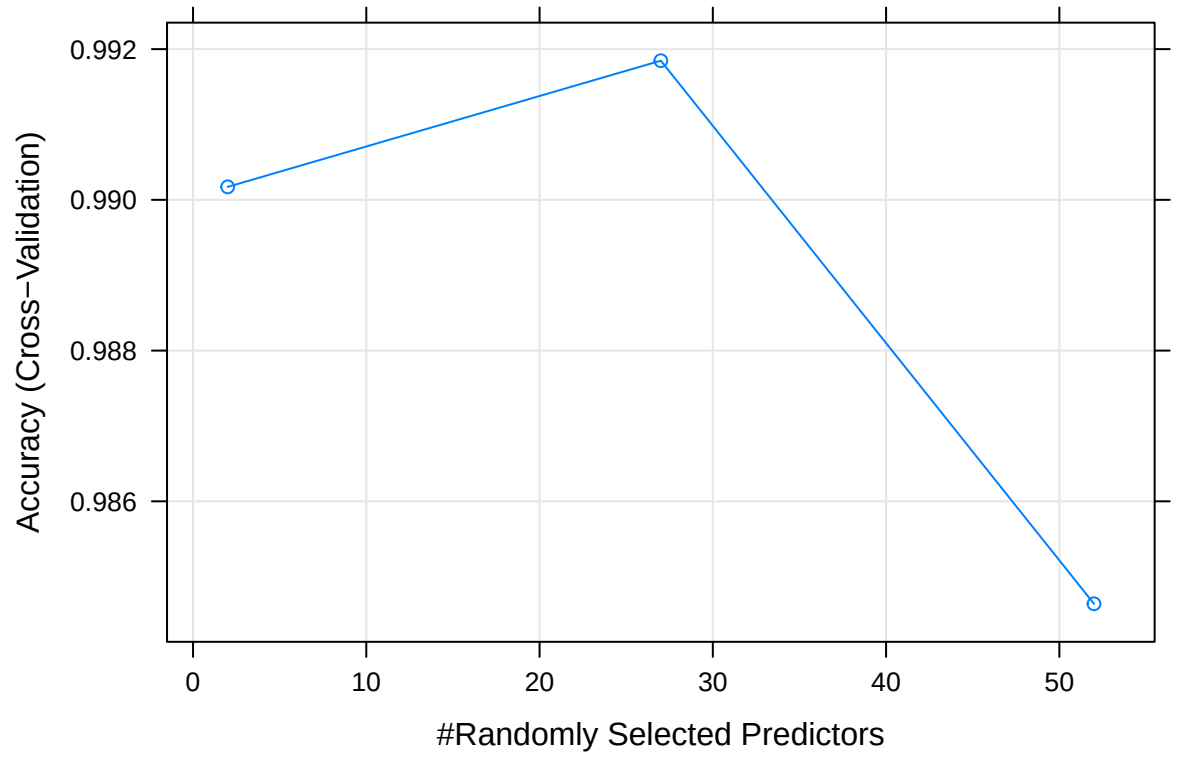
```
# Model fit
set.seed(1234)
RF <- train(  classe ~ .,   data=training,  trControl=fitControl,  method='rf',  ntree=100)
save(RF, file='./ModelFitRF.RData')

# Prediction on validation data set
predRF <- predict(RF, newdata=validity)
cmRF <-confusionMatrix(predRF, factor(validity$classe))
cmRF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    4    0    0    0
##          B    1 1131   13    0    1
##          C    1    4 1011    7    0
##          D    0    0    2  956    0
##          E    0    0    0    1 1081
##
## Overall Statistics
##
```

```
##                Accuracy : 0.9942
##                  95% CI : (0.9919, 0.996)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9927
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9930   0.9854   0.9917   0.9991
## Specificity            0.9991   0.9968   0.9975   0.9996   0.9998
## Pos Pred Value         0.9976   0.9869   0.9883   0.9979   0.9991
## Neg Pred Value         0.9995   0.9983   0.9969   0.9984   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2841   0.1922   0.1718   0.1624   0.1837
## Detection Prevalence   0.2848   0.1947   0.1738   0.1628   0.1839
## Balanced Accuracy      0.9989   0.9949   0.9915   0.9956   0.9994
```

```r
# Plotting the model
plot(RF)
```



d) Support Vector Machine Model

```
# Model fit
set.seed(1234)
SVM <- train(  classe ~ .,   data=training,  trControl=fitControl,  method='svmLinear')
save(SVM, file='./ModelFitSVM.RData')

# Prediction on validation data set
predSVM <- predict(SVM, newdata=validity)
cmSVM <-confusionMatrix(predSVM, factor(validity$classe))
cmSVM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1537  154   79   69   50
##          B   29  806   90   46  152
##          C   40   81  797  114   69
##          D   61   22   32  697   50
##          E    7   76   28   38  761
##
## Overall Statistics
##
##                Accuracy : 0.7813
##                  95% CI : (0.7705, 0.7918)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.722
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9182   0.7076   0.7768   0.7230   0.7033
## Specificity            0.9164   0.9332   0.9374   0.9665   0.9690
## Pos Pred Value         0.8137   0.7177   0.7239   0.8086   0.8363
## Neg Pred Value         0.9657   0.9301   0.9521   0.9468   0.9355
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2612   0.1370   0.1354   0.1184   0.1293
## Detection Prevalence   0.3210   0.1908   0.1871   0.1465   0.1546
## Balanced Accuracy      0.9173   0.8204   0.8571   0.8447   0.8362
```

Results- Accuracy and out of sample error:

```
accuracyResults <- data.frame(Mod = c('RPART', 'GBM', 'RF', 'SVM'),
  Accuracy = rbind(cmRPART$overall[1], cmGBM$overall[1], cmRF$overall[1], cmSVM$overall[1])
)
print(accuracyResults)
```

```
##     Mod  Accuracy
## 1 RPART 0.4937978
## 2   GBM 0.9670348
## 3    RF 0.9942226
## 4   SVM 0.7813084
```

Of the four models (RPART, SDM,RF & SVM) used, the random forest (RF) was the model out performed and being the best with accuracy of 0.9943925 and out of sample error of 0.0056075. RF model can be used to perform in the test data.

Applying the best model (RF) to the Test data set:

```
predTest <- predict(RF, test )

print(predTest)
```

```
##  [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
```