Coursera Practical machine learning programming assignment

Peter B.

17 Juni 2018

Synopsis

Background information and assignment instructions

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. 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).

Data

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)

Data source: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) [last accessed: 15June2018] more details can be found here: Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

Assigment

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.

Summary

After reading the datasets a basic exploratory data analyses is performed. In a next step empty variables are excluded as well as variables with a zero or close to zero variance that won't conribute to a model.The training data is for cross-validation purposes partioned in a test and training data set. In a first step a decision tree model is applied. The accurany with 0.7 is not conincing. So a random forest model is used. Here an accuracy of 0.98 is obtained. The expected out of sample rate is 2.3% and acceptable.

Programming part

Getting the data

```
#fileUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
#download.file(fileUrl, destfile= paste0(getwd(), "pml-training.csv"))
training0 <- read.csv(paste0(getwd(), "pml-training.csv"))

#fileUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
#download.file(fileUrl, destfile= paste0(getwd(), "pml-testing.csv"))
testing0 <- read.csv(paste0(getwd(), "pml-testing.csv"))
```

Explorative data analyses

To get an first impression of the available data explorative data analyses tools are used to summarize the data and get information and variable types etc. For readablity purposes only the codeis shown in the submission.

```
# str(training0)
# dim(training0)
# summary(training0)
```

As we have to deal in this dataset with a lot of missing values they will now be analyzed further using the testing data set, because variables that here are completely missing do not contribute to a model.

Cleaning data

In this part of the assignment - all empty and not model suited variables are removed. The variables that are not suitable in the modeling are: “X”, “user_name”, “raw_timestamp_part_1”, “raw_timestamp_part_2”, “cvtd_timestamp”, “new_window”, “num_window” and the variables that are empty are statistics related (mean, standard deviation, skewness, kurtosis and variance). So all variables that are usable will be kept and in a next step those variables will be kept in both - training and testing - data sets. - in a second step all variables that do not conribute based on a varianve close to zero or zero will be removed as they do also not conribute to a model fitting using the nearZeroVar function from the caret package.

```r
# remove variables with only missing values
colSums(is.na(training0))
```

```
##                      X              user_name   raw_timestamp_part_1
##                      0                      0                      0
##   raw_timestamp_part_2         cvtd_timestamp             new_window
##                      0                      0                      0
##             num_window              roll_belt             pitch_belt
##                      0                      0                      0
##               yaw_belt       total_accel_belt     kurtosis_roll_belt
##                      0                      0                      0
##    kurtosis_picth_belt      kurtosis_yaw_belt     skewness_roll_belt
##                      0                      0                      0
##   skewness_roll_belt.1      skewness_yaw_belt          max_roll_belt
##                      0                      0                  19216
##         max_picth_belt           max_yaw_belt          min_roll_belt
##                  19216                      0                  19216
##         min_pitch_belt           min_yaw_belt    amplitude_roll_belt
##                  19216                      0                  19216
##   amplitude_pitch_belt     amplitude_yaw_belt   var_total_accel_belt
##                  19216                      0                  19216
##          avg_roll_belt       stddev_roll_belt          var_roll_belt
##                  19216                  19216                  19216
##         avg_pitch_belt      stddev_pitch_belt         var_pitch_belt
##                  19216                  19216                  19216
##           avg_yaw_belt        stddev_yaw_belt           var_yaw_belt
##                  19216                  19216                  19216
##           gyros_belt_x           gyros_belt_y           gyros_belt_z
##                      0                      0                      0
##           accel_belt_x           accel_belt_y           accel_belt_z
##                      0                      0                      0
##          magnet_belt_x          magnet_belt_y          magnet_belt_z
##                      0                      0                      0
##               roll_arm              pitch_arm                yaw_arm
##                      0                      0                      0
##        total_accel_arm          var_accel_arm           avg_roll_arm
##                      0                  19216                  19216
##        stddev_roll_arm           var_roll_arm          avg_pitch_arm
##                  19216                  19216                  19216
##       stddev_pitch_arm          var_pitch_arm            avg_yaw_arm
##                  19216                  19216                  19216
##         stddev_yaw_arm            var_yaw_arm            gyros_arm_x
##                  19216                  19216                      0
##            gyros_arm_y            gyros_arm_z            accel_arm_x
##                      0                      0                      0
##            accel_arm_y            accel_arm_z           magnet_arm_x
##                      0                      0                      0
##           magnet_arm_y           magnet_arm_z      kurtosis_roll_arm
##                      0                      0                      0
##     kurtosis_picth_arm       kurtosis_yaw_arm      skewness_roll_arm
##                      0                      0                      0
##     skewness_pitch_arm       skewness_yaw_arm           max_roll_arm
##                      0                      0                  19216
##          max_picth_arm            max_yaw_arm           min_roll_arm
##                  19216                  19216                  19216
```

```
##              min_pitch_arm              min_yaw_arm        amplitude_roll_arm 
##                      19216                    19216                     19216 
##         amplitude_pitch_arm        amplitude_yaw_arm             roll_dumbbell 
##                      19216                    19216                         0 
##             pitch_dumbbell             yaw_dumbbell    kurtosis_roll_dumbbell 
##                          0                        0                         0 
##    kurtosis_picth_dumbbell    kurtosis_yaw_dumbbell    skewness_roll_dumbbell 
##                          0                        0                         0 
##    skewness_pitch_dumbbell    skewness_yaw_dumbbell         max_roll_dumbbell 
##                          0                        0                     19216 
##         max_picth_dumbbell         max_yaw_dumbbell         min_roll_dumbbell 
##                      19216                        0                     19216 
##         min_pitch_dumbbell         min_yaw_dumbbell   amplitude_roll_dumbbell 
##                      19216                        0                     19216 
##   amplitude_pitch_dumbbell   amplitude_yaw_dumbbell      total_accel_dumbbell 
##                      19216                        0                         0 
##         var_accel_dumbbell        avg_roll_dumbbell      stddev_roll_dumbbell 
##                      19216                    19216                     19216 
##          var_roll_dumbbell       avg_pitch_dumbbell     stddev_pitch_dumbbell 
##                      19216                    19216                     19216 
##         var_pitch_dumbbell         avg_yaw_dumbbell       stddev_yaw_dumbbell 
##                      19216                    19216                     19216 
##           var_yaw_dumbbell         gyros_dumbbell_x          gyros_dumbbell_y 
##                      19216                        0                         0 
##           gyros_dumbbell_z         accel_dumbbell_x          accel_dumbbell_y 
##                          0                        0                         0 
##           accel_dumbbell_z        magnet_dumbbell_x         magnet_dumbbell_y 
##                          0                        0                         0 
##          magnet_dumbbell_z             roll_forearm             pitch_forearm 
##                          0                        0                         0 
##                yaw_forearm    kurtosis_roll_forearm    kurtosis_picth_forearm 
##                          0                        0                         0 
##       kurtosis_yaw_forearm    skewness_roll_forearm    skewness_pitch_forearm 
##                          0                        0                         0 
##       skewness_yaw_forearm         max_roll_forearm         max_picth_forearm 
##                          0                    19216                     19216 
##            max_yaw_forearm         min_roll_forearm         min_pitch_forearm 
##                          0                    19216                     19216 
##            min_yaw_forearm   amplitude_roll_forearm   amplitude_pitch_forearm 
##                          0                    19216                     19216 
##      amplitude_yaw_forearm        total_accel_forearm         var_accel_forearm 
##                          0                        0                     19216 
##           avg_roll_forearm       stddev_roll_forearm          var_roll_forearm 
##                      19216                    19216                     19216 
##          avg_pitch_forearm      stddev_pitch_forearm         var_pitch_forearm 
##                      19216                    19216                     19216 
##            avg_yaw_forearm        stddev_yaw_forearm           var_yaw_forearm 
##                      19216                    19216                     19216 
##            gyros_forearm_x          gyros_forearm_y           gyros_forearm_z 
##                          0                        0                         0 
##            accel_forearm_x          accel_forearm_y           accel_forearm_z 
##                          0                        0                         0 
```

```
##        magnet_forearm_x        magnet_forearm_y        magnet_forearm_z
##                       0                       0                       0
##                  classe
##                       0
```

```
keepVar <- names(training0[,colSums(is.na(training0)) == 0])
keepVar2 <- keepVar[8:59]

training1 <- training0[,c(keepVar2,"classe")]
testing1  <- testing0[,c(keepVar2,"problem_id")]

# remove variables with a variance close to zero
NZV <- nearZeroVar(training1)
training2 <- training1[, -NZV]
testingFinal  <- testing1[, -NZV]
dim(training2); dim(testingFinal)
```

```
## [1] 19622    30
```

```
## [1] 20 30
```

Using this two approaches the input dataset for the modeling is reduced from 160 variables to 30 variables.

Partioning the data

The “training” data set will be split into a training data set containing 60% of the observations and a testing data set (40% of the total cases) bases on the outcome variable “classe”. This will allow us to perform a cross-validation and estimate the out of sample error. For reproducibilty purposes we will also set a seed, here: 999.

```
set.seed(999)
library(caret)
inTrain  <- createDataPartition(training2$classe, p=0.6, list=FALSE)
training <- training2[inTrain,]
testing  <- training2[-inTrain,]
dim(training);dim(testing)
```

```
## [1] 11776    30
```

```
## [1] 7846   30
```

Correlation

In order to identify potential confounding issues the correlations among the variables are examined.

```
corrMatrix <- cor(training2[, -30])
```

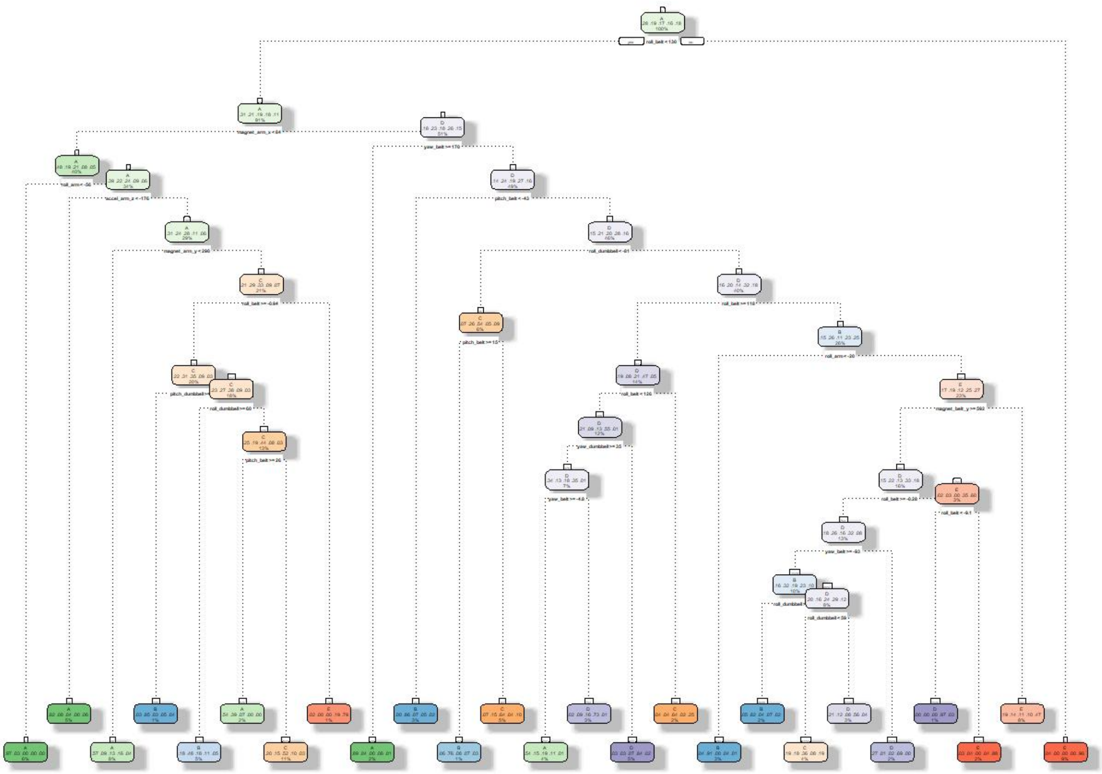
Examing the correlations among the predictors the majority is unrelated and so we continue without further pre-processing of the data.

prediction modelling

In this step a decision tree as starting point is used and relevant statistics examined

```
set.seed(999)
library(rpart)
library(rpart.plot)
modFit <- rpart(classe ~ ., data = training, method="class")
fancyRpartPlot(modFit)
```

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



Rattle 2018-Jun-17 18:27:09 bonat

```
set.seed(999)
prediction <- predict(modFit, testing, type = "class")
confusionMatrix(prediction, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1598  218  183  144   65
##          B   88  835   95   83   46
##          C  261  286  947  163  157
##          D  129   84   85  824   26
##          E  156   95   58   72 1148
##
## Overall Statistics
##
##                Accuracy : 0.6821
##                  95% CI : (0.6717, 0.6924)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5984
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7159   0.5501   0.6923   0.6407   0.7961
## Specificity            0.8913   0.9507   0.8662   0.9506   0.9405
## Pos Pred Value         0.7237   0.7280   0.5221   0.7178   0.7508
## Neg Pred Value         0.8875   0.8980   0.9302   0.9310   0.9535
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2037   0.1064   0.1207   0.1050   0.1463
## Detection Prevalence   0.2814   0.1462   0.2312   0.1463   0.1949
## Balanced Accuracy      0.8036   0.7504   0.7792   0.7957   0.8683
```

A Random Forest model is created. Here a control dataset is used with the method cv and the number of resampling is set to three.

```
#set.seed(999)
#modFitRF <- randomForest(classe ~ ., data = training, ntree = 500)
#prediction <- predict(modFitRF, testing, type = "class")
#confusionMatrix(prediction, testing$classe)



set.seed(999)
controlRf <- trainControl(method="cv", number=3, verboseIter=FALSE)
modFitRF2<- train(classe ~ ., data=training, method="rf",
                          trControl=controlRf)
modFitRF2$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: 2
##
##         OOB estimate of  error rate: 2.32%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3299   12   15   21    1  0.01463560
## B   37 2211   25    5    1  0.02983765
## C    6   41 1994   12    1  0.02921130
## D    9    2   52 1860    7  0.03626943
## E    1    7   12    6 2139  0.01200924
```

```
# prediction on Test dataset
set.seed(999)
predictRF <- predict(modFitRF2, newdata=testing)
confusionMatRF <- confusionMatrix(predictRF, testing$classe)
confusionMatRF
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 2199   26    4    2    5
##          B    7 1474   14    0   10
##          C    9   13 1341   48    7
##          D   17    5    9 1236    2
##          E    0    0    0    0 1418
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9773          
##                  95% CI : (0.9738, 0.9805)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9713          
##  Mcnemar's Test P-Value : 4.142e-13       
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9852   0.9710   0.9803   0.9611   0.9834
## Specificity            0.9934   0.9951   0.9881   0.9950   1.0000
## Pos Pred Value         0.9835   0.9794   0.9457   0.9740   1.0000
## Neg Pred Value         0.9941   0.9931   0.9958   0.9924   0.9963
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2803   0.1879   0.1709   0.1575   0.1807
## Detection Prevalence   0.2850   0.1918   0.1807   0.1617   0.1807
## Balanced Accuracy      0.9893   0.9831   0.9842   0.9780   0.9917
```

Using the test data set for cross- validation an accuracy of 0.98 is obtained. Though there might be still a tendency to overfitting, this model will be used as final model. The expected out of sample error is 2,32 %.

Conclusion

See summary at the beginning of the document.

Applying Random forest predcition to the provided 20 test cases

```
predictSubm <- predict(modFitRF2, newdata=testingFinal)
predictSubm
```

```
##  [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
```