# Course Project

Andrés Ramírez

3/1/2021

Background Subjects were asked to perform barbell lifts correctly and incorrectly in 5 different ways. * Exactly according to the specification (Class A) * Throwing the elbows to the front (Class B) - mistake * Lifting the dumbbell only halfway (Class C) - mistake * Lowering the dumbbell only halfway (Class D) - mistake * Throwing the hips to the front (Class E) - mistake

Accelerometers were located on 1. belt 2. forearm 3. arm

Task Create a report describing * how you built your model, * how you used cross validation * what you think the expected out of sample error is * why you made the choices you did

Overview The model building workflow adopted for this task follows the pattern outlined in lectures:

```
> question .. input .. features .. algorithm .. predict .. evaluation
```

Cross Validation has been used as a method for the trainControl function with 4 folds used.

The out of sample error was found to be 0.0037% when the model was applied to the test data derived from the training set.

Choices made at each step are described in the workflow below.

Setup Due to size of the training sample (19622 observations and up to 60 variables), parallel processing was selected for model development

```
suppressWarnings(suppressMessages(library(caret)))
suppressWarnings(suppressMessages(library(randomForest)))
suppressWarnings(suppressMessages(library(e1071)))
set.seed(1603)
```

QUESTION Create a model to predict the manner in which the subjects did the exercise using the accelerometer data as predictors. The outcome to be predicted is the “classe” variable.

INPUT Download source data

```
trainingFilename   <- 'pml-training.csv'
quizFilename       <- 'pml-testing.csv'
trainingUrl        <-
'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
```

```
quizUrl            <-
'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
```

Data Cleansing On inspection in Excel, found NA,#DIV/0! and blank values in the data. These are not valid observed values, so remove with na.strings parameter.

```
training.df     <-read.csv(trainingFilename,
na.strings=c("NA","","#DIV/0!"))
training.df     <-training.df[,colSums(is.na(training.df)) == 0]
dim(training.df) #;head(training.df,3)

## [1] 19622    60

quiz.df         <-read.csv(quizFilename , na.strings=c("NA", "",
"#DIV/0!"))
quiz.df         <-quiz.df[,colSums(is.na(quiz.df)) == 0]
dim(quiz.df) #;head(quiz.df,3)

## [1] 20 60
```

FEATURES Reduce the number of variables Remove the non-predictors from the training set. This includes the index, subject name, time and window variables.

```
Training.df   <-training.df[,-c(1:7)]
Quiz.df <-quiz.df[,-c(1:7)]
dim(Training.df)

## [1] 19622    53
```

Check for near zero values in training data

```
Training.nzv<-nzv(Training.df[,-ncol(Training.df)],saveMetrics=TRUE)
```

None found so display and count variables submitted for the train function

```
rownames(Training.nzv)

##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"
"total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
```

```
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"

dim(Training.nzv)[1]

## [1] 52
```

ALGORITHM Partition the training data into a training set and a testing/validation set

```
inTrain     <- createDataPartition(Training.df$classe, p = 0.6, list =
FALSE)
inTraining  <- Training.df[inTrain,]
inTest      <- Training.df[-inTrain,]
dim(inTraining);dim(inTest)

## [1] 11776    53

## [1] 7846   53
```

Construct the model using cross validation or reload using the cached model Cross Validation achieved with trainControl method set to “cv”

```
myModelFilename <- "myModel.RData"
if (!file.exists(myModelFilename)) {

    # Parallel cores
    #require(parallel)
    library(doParallel)
    ncores <- makeCluster(detectCores() - 1)
    registerDoParallel(cores=ncores)
    getDoParWorkers() # 3

    # use Random Forest method with Cross Validation, 4 folds
    myModel <- train(classe ~ .
                , data = inTraining
                , method = "rf"
                , metric = "Accuracy"  # categorical outcome variable so
choose accuracy
                , preProcess=c("center", "scale") # attempt to improve
accuracy by normalising
                , trControl=trainControl(method = "cv"
                                         , number = 4 # folds of the
training data
                                         , p= 0.60
                                         , allowParallel = TRUE
#                                        , seeds=NA # don't let workers
set seed
                                         )
                )
```

```
    save(myModel, file = "myModel.RData")
    # 3:42 .. 3:49 without preProcess
    # 3:51 .. 3:58 with preProcess
    stopCluster(ncores)
} else {
    # Use cached model
    load(file = myModelFilename, verbose = TRUE)
}
```

```
## Loading objects:
##   myModel
```

```
print(myModel, digits=4)
```

```
## Random Forest
##
## 11776 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## Pre-processing: centered (52), scaled (52)
## Resampling: Cross-Validated (4 fold)
## Summary of sample sizes: 8831, 8831, 8834, 8832
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa
##    2    0.9872    0.9838
##   27    0.9882    0.9851
##   52    0.9817    0.9768
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

PREDICT

Predicting the activity performed using the training file derived test subset

```
predTest <- predict(myModel, newdata=inTest)
```

Final Model data and important predictors in the model

```
myModel$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: 0.86%
## Confusion matrix:
```

```
##      A    B    C    D    E class.error
## A 3344    2    1    0    1 0.001194743
## B   20 2251    6    2    0 0.012286090
## C    0   14 2031    9    0 0.011197663
## D    0    1   29 1897    3 0.017098446
## E    0    2    4    7 2152 0.006004619
```

```
varImp(myModel)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 52)
##
##                      Overall
## roll_belt            100.000
## pitch_forearm         61.314
## yaw_belt              54.425
## pitch_belt            44.937
## magnet_dumbbell_z     42.705
## magnet_dumbbell_y     42.677
## roll_forearm          40.036
## accel_dumbbell_y      23.081
## magnet_dumbbell_x     18.778
## roll_dumbbell         18.585
## accel_forearm_x       16.913
## magnet_belt_z         16.052
## accel_dumbbell_z      14.119
## magnet_forearm_z      13.891
## magnet_belt_y         13.496
## total_accel_dumbbell  12.884
## accel_belt_z          12.413
## gyros_belt_z          11.382
## yaw_arm               10.311
## magnet_belt_x          9.237
```

27 variables were tried at each split and the reported OOB Estimated Error is a low 0.83%.

Overall we have sufficient confidence in the prediction model to predict classe for the 20 quiz/test cases.

Validation/Quiz The accuracy of the model by predicting with the Validation/Quiz set supplied in the test file.

```
print(predict(myModel, newdata=Quiz.df))
```

```
##  [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
```