John Hopkins University – Data Science Specialization – Practical Machine Learning Course – Project 1

Dr. Guy Cohen

August 21, 2015

Human Activity Recognition - Weight Lifting Data

Problem

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 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 (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

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

What you should submit

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.

1. Your submission 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 :-).
2. You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

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.

Solution

I verified the length of the text (not including code and code output) to be less than 2,000 words. The number of figures is less than 5.

Getting and Cleaning the Data

```
set.seed(33833)
data1 <- read.table("pml-training.csv", header = TRUE, sep=",",
                    stringsAsFactors = FALSE, na.strings = c("","NA"))
```

Remove serial number, time stamps and window information features. Remove features which have more than 50% missing values.

```
data1 <- data1[, -c(1,3,4,5,6,7)]
incompleteFeatures <- which((sapply(data1, function(x) {
    sum(is.na(x))}) / dim(data1)[1]) > 0.5)
data1 <- data1[, -incompleteFeatures]
print(paste("Percent of complete cases is now: ",
            sum(complete.cases(data1)) / dim(data1)[1] * 100,"%", sep = ""))
```

```
## [1] "Percent of complete cases is now: 100%"
```

```
print(paste("Number of features is now: ",dim(data1)[2], sep = ""))
```

```
## [1] "Number of features is now: 54"
```

Convert "user_name" and "classe" features to factor variables.

```
data1[,c("user_name","classe")] <-
    as.data.frame(lapply(data1[,c("user_name","classe")], factor))
```

Exploratory Data Analysis

We examine the structure and the summary of the features.

```
str(data1)
```

```
## 'data.frame':    19622 obs. of  54 variables:
##  $ user_name           : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ roll_belt           : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...
##  $ pitch_belt          : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...
```

```
##  $ yaw_belt            : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...
##  $ total_accel_belt    : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ gyros_belt_x        : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 0.03 ...
##  $ gyros_belt_y        : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z        : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02 -0.02 0 ...
##  $ accel_belt_x        : int  -21 -22 -20 -22 -21 -21 -22 -22 -20 -21 ...
##  $ accel_belt_y        : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z        : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x       : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
##  $ magnet_belt_y       : int  599 608 600 604 600 603 599 603 602 609 ...
##  $ magnet_belt_z       : int  -313 -311 -305 -310 -302 -312 -311 -313 -312 -308 ...
##  $ roll_arm            : num  -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...
##  $ pitch_arm           : num  22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...
##  $ yaw_arm             : num  -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...
##  $ total_accel_arm     : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ gyros_arm_x         : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 ...
##  $ gyros_arm_y         : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.02 -0.03 -0.03 ...
##  $ gyros_arm_z         : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02 -0.02 ...
##  $ accel_arm_x         : int  -288 -290 -289 -289 -289 -289 -289 -289 -288 -288 ...
##  $ accel_arm_y         : int  109 110 110 111 111 111 111 111 109 110 ...
##  $ accel_arm_z         : int  -123 -125 -126 -123 -123 -122 -125 -124 -122 -124 ...
##  $ magnet_arm_x        : int  -368 -369 -368 -372 -374 -369 -373 -372 -369 -376 ...
##  $ magnet_arm_y        : int  337 337 344 344 337 342 336 338 341 334 ...
##  $ magnet_arm_z        : int  516 513 513 512 506 513 509 510 518 516 ...
##  $ roll_dumbbell       : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell      : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell        : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ total_accel_dumbbell: int  37 37 37 37 37 37 37 37 37 37 ...
##  $ gyros_dumbbell_x    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ gyros_dumbbell_y    : num  -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 ...
##  $ gyros_dumbbell_z    : num  0 0 0 -0.02 0 0 0 0 0 0 ...
##  $ accel_dumbbell_x    : int  -234 -233 -232 -232 -233 -234 -232 -234 -232 -235 ...
##  $ accel_dumbbell_y    : int  47 47 46 48 48 48 47 46 47 48 ...
##  $ accel_dumbbell_z    : int  -271 -269 -270 -269 -270 -269 -270 -272 -269 -270 ...
##  $ magnet_dumbbell_x   : int  -559 -555 -561 -552 -554 -558 -551 -555 -549 -558 ...
##  $ magnet_dumbbell_y   : int  293 296 298 303 292 294 295 300 292 291 ...
##  $ magnet_dumbbell_z   : num  -65 -64 -63 -60 -68 -66 -70 -74 -65 -69 ...
##  $ roll_forearm        : num  28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...
##  $ yaw_forearm         : num  -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...
##  $ total_accel_forearm : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ gyros_forearm_x     : num  0.03 0.02 0.03 0.02 0.02 0.02 0.02 0.02 0.03 0.02 ...
##  $ gyros_forearm_y     : num  0 0 -0.02 -0.02 0 -0.02 0 -0.02 0 0 ...
##  $ gyros_forearm_z     : num  -0.02 -0.02 0 0 -0.02 -0.03 -0.02 0 -0.02 -0.02 ...
##  $ accel_forearm_x     : int  192 192 196 189 189 193 195 193 193 190 ...
##  $ accel_forearm_y     : int  203 203 204 206 206 203 205 205 204 205 ...
##  $ accel_forearm_z     : int  -215 -216 -213 -214 -214 -215 -215 -213 -214 -215 ...
##  $ magnet_forearm_x    : int  -17 -18 -18 -16 -17 -9 -18 -9 -16 -22 ...
##  $ magnet_forearm_y    : num  654 661 658 658 655 660 659 660 653 656 ...
##  $ magnet_forearm_z    : num  476 473 469 469 473 478 470 474 476 473 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```r
summary(data1)
```

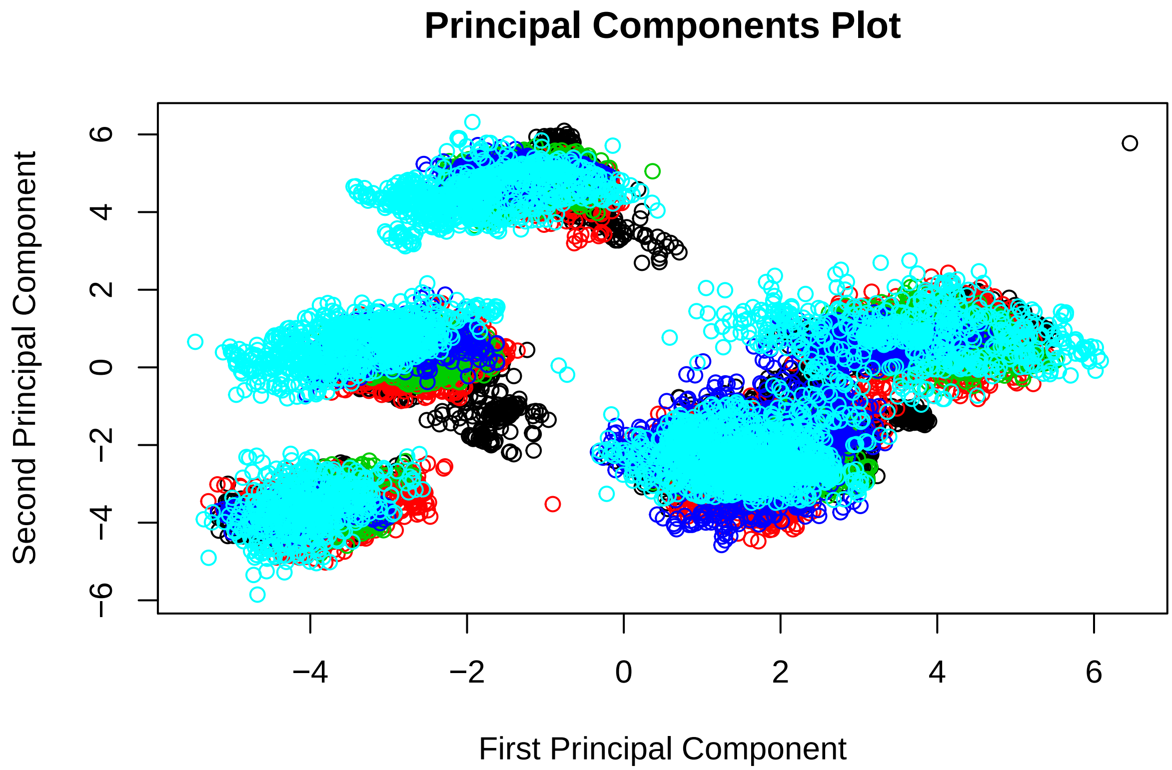
```
##    user_name      roll_belt        pitch_belt          yaw_belt
##  adelmo  :3892   Min.   :-28.90   Min.   :-55.8000   Min.   :-180.00
##  carlitos:3112   1st Qu.:  1.10   1st Qu.:  1.7600   1st Qu.: -88.30
##  charles :3536   Median :113.00   Median :  5.2800   Median : -13.00
##  eurico  :3070   Mean   : 64.41   Mean   :  0.3053   Mean   : -11.21
##  jeremy  :3402   3rd Qu.:123.00   3rd Qu.: 14.9000   3rd Qu.:  12.90
##  pedro   :2610   Max.   :162.00   Max.   : 60.3000   Max.   : 179.00
##  total_accel_belt  gyros_belt_x        gyros_belt_y       gyros_belt_z
##  Min.   : 0.00    Min.   :-1.040000   Min.   :-0.64000   Min.   :-1.4600
##  1st Qu.: 3.00    1st Qu.:-0.030000   1st Qu.: 0.00000   1st Qu.:-0.2000
##  Median :17.00    Median : 0.030000   Median : 0.02000   Median :-0.1000
##  Mean   :11.31    Mean   :-0.005592   Mean   : 0.03959   Mean   :-0.1305
##  3rd Qu.:18.00    3rd Qu.: 0.110000   3rd Qu.: 0.11000   3rd Qu.:-0.0200
##  Max.   :29.00    Max.   : 2.220000   Max.   : 0.64000   Max.   : 1.6200
##   accel_belt_x       accel_belt_y     accel_belt_z     magnet_belt_x
##  Min.   :-120.000   Min.   :-69.00   Min.   :-275.00   Min.   :-52.0
##  1st Qu.: -21.000   1st Qu.:  3.00   1st Qu.:-162.00   1st Qu.:  9.0
##  Median : -15.000   Median : 35.00   Median :-152.00   Median : 35.0
##  Mean   :  -5.595   Mean   : 30.15   Mean   : -72.59   Mean   : 55.6
##  3rd Qu.:  -5.000   3rd Qu.: 61.00   3rd Qu.:  27.00   3rd Qu.: 59.0
##  Max.   :  85.000   Max.   :164.00   Max.   : 105.00   Max.   :485.0
##  magnet_belt_y   magnet_belt_z       roll_arm         pitch_arm
##  Min.   :354.0   Min.   :-623.0   Min.   :-180.00   Min.   :-88.800
##  1st Qu.:581.0   1st Qu.:-375.0   1st Qu.: -31.77   1st Qu.:-25.900
##  Median :601.0   Median :-320.0   Median :   0.00   Median :  0.000
##  Mean   :593.7   Mean   :-345.5   Mean   :  17.83   Mean   : -4.612
##  3rd Qu.:610.0   3rd Qu.:-306.0   3rd Qu.:  77.30   3rd Qu.: 11.200
##  Max.   :673.0   Max.   : 293.0   Max.   : 180.00   Max.   : 88.500
##     yaw_arm          total_accel_arm  gyros_arm_x       gyros_arm_y
##  Min.   :-180.0000   Min.   : 1.00   Min.   :-6.37000   Min.   :-3.4400
##  1st Qu.: -43.1000   1st Qu.:17.00   1st Qu.:-1.33000   1st Qu.:-0.8000
##  Median :   0.0000   Median :27.00   Median : 0.08000   Median :-0.2400
##  Mean   :  -0.6188   Mean   :25.51   Mean   : 0.04277   Mean   :-0.2571
##  3rd Qu.:  45.8750   3rd Qu.:33.00   3rd Qu.: 1.57000   3rd Qu.: 0.1400
##  Max.   : 180.0000   Max.   :66.00   Max.   : 4.87000   Max.   : 2.8400
##   gyros_arm_z       accel_arm_x       accel_arm_y      accel_arm_z
##  Min.   :-2.3300   Min.   :-404.00   Min.   :-318.0   Min.   :-636.00
##  1st Qu.:-0.0700   1st Qu.:-242.00   1st Qu.: -54.0   1st Qu.:-143.00
##  Median : 0.2300   Median : -44.00   Median :  14.0   Median : -47.00
##  Mean   : 0.2695   Mean   : -60.24   Mean   :  32.6   Mean   : -71.25
##  3rd Qu.: 0.7200   3rd Qu.:  84.00   3rd Qu.: 139.0   3rd Qu.:  23.00
##  Max.   : 3.0200   Max.   : 437.00   Max.   : 308.0   Max.   : 292.00
##   magnet_arm_x     magnet_arm_y     magnet_arm_z    roll_dumbbell
##  Min.   :-584.0   Min.   :-392.0   Min.   :-597.0   Min.   :-153.71
##  1st Qu.:-300.0   1st Qu.:  -9.0   1st Qu.: 131.2   1st Qu.: -18.49
##  Median : 289.0   Median : 202.0   Median : 444.0   Median :  48.17
##  Mean   : 191.7   Mean   : 156.6   Mean   : 306.5   Mean   :  23.84
##  3rd Qu.: 637.0   3rd Qu.: 323.0   3rd Qu.: 545.0   3rd Qu.:  67.61
##  Max.   : 782.0   Max.   : 583.0   Max.   : 694.0   Max.   : 153.55
##  pitch_dumbbell     yaw_dumbbell      total_accel_dumbbell
##  Min.   :-149.59   Min.   :-150.871   Min.   : 0.00
##  1st Qu.: -40.89   1st Qu.: -77.644   1st Qu.: 4.00
##  Median : -20.96   Median :  -3.324   Median :10.00
##  Mean   : -10.78   Mean   :   1.674   Mean   :13.72
```

```
##  3rd Qu.:  17.50   3rd Qu.:  79.643   3rd Qu.:19.00
##  Max.   : 149.40   Max.   : 154.952   Max.   :58.00
##  gyros_dumbbell_x    gyros_dumbbell_y   gyros_dumbbell_z
##  Min.   :-204.0000   Min.   :-2.10000   Min.   : -2.380
##  1st Qu.:  -0.0300   1st Qu.:-0.14000   1st Qu.: -0.310
##  Median :   0.1300   Median : 0.03000   Median : -0.130
##  Mean   :   0.1611   Mean   : 0.04606   Mean   : -0.129
##  3rd Qu.:   0.3500   3rd Qu.: 0.21000   3rd Qu.:  0.030
##  Max.   :   2.2200   Max.   :52.00000   Max.   :317.000
##  accel_dumbbell_x  accel_dumbbell_y  accel_dumbbell_z  magnet_dumbbell_x
##  Min.   :-419.00   Min.   :-189.00   Min.   :-334.00   Min.   :-643.0
##  1st Qu.: -50.00   1st Qu.:  -8.00   1st Qu.:-142.00   1st Qu.:-535.0
##  Median :  -8.00   Median :  41.50   Median :  -1.00   Median :-479.0
##  Mean   : -28.62   Mean   :  52.63   Mean   : -38.32   Mean   :-328.5
##  3rd Qu.:  11.00   3rd Qu.: 111.00   3rd Qu.:  38.00   3rd Qu.:-304.0
##  Max.   : 235.00   Max.   : 315.00   Max.   : 318.00   Max.   : 592.0
##  magnet_dumbbell_y magnet_dumbbell_z  roll_forearm       pitch_forearm
##  Min.   :-3600     Min.   :-262.00   Min.   :-180.0000   Min.   :-72.50
##  1st Qu.:  231     1st Qu.: -45.00   1st Qu.:  -0.7375   1st Qu.:  0.00
##  Median :  311     Median :  13.00   Median :  21.7000   Median :  9.24
##  Mean   :  221     Mean   :  46.05   Mean   :  33.8265   Mean   : 10.71
##  3rd Qu.:  390     3rd Qu.:  95.00   3rd Qu.: 140.0000   3rd Qu.: 28.40
##  Max.   :  633     Max.   : 452.00   Max.   : 180.0000   Max.   : 89.80
##   yaw_forearm      total_accel_forearm gyros_forearm_x
##  Min.   :-180.00   Min.   :  0.00      Min.   :-22.000
##  1st Qu.: -68.60   1st Qu.: 29.00      1st Qu.: -0.220
##  Median :   0.00   Median : 36.00      Median :  0.050
##  Mean   :  19.21   Mean   : 34.72      Mean   :  0.158
##  3rd Qu.: 110.00   3rd Qu.: 41.00      3rd Qu.:  0.560
##  Max.   : 180.00   Max.   :108.00      Max.   :  3.970
##  gyros_forearm_y     gyros_forearm_z    accel_forearm_x   accel_forearm_y
##  Min.   : -7.02000   Min.   : -8.0900   Min.   :-498.00   Min.   :-632.0
##  1st Qu.: -1.46000   1st Qu.: -0.1800   1st Qu.:-178.00   1st Qu.:  57.0
##  Median :  0.03000   Median :  0.0800   Median : -57.00   Median : 201.0
##  Mean   :  0.07517   Mean   :  0.1512   Mean   : -61.65   Mean   : 163.7
##  3rd Qu.:  1.62000   3rd Qu.:  0.4900   3rd Qu.:  76.00   3rd Qu.: 312.0
##  Max.   :311.00000   Max.   :231.0000   Max.   : 477.00   Max.   : 923.0
##  accel_forearm_z   magnet_forearm_x  magnet_forearm_y magnet_forearm_z
##  Min.   :-446.00   Min.   :-1280.0   Min.   :-896.0   Min.   :-973.0
##  1st Qu.:-182.00   1st Qu.: -616.0   1st Qu.:   2.0   1st Qu.: 191.0
##  Median : -39.00   Median : -378.0   Median : 591.0   Median : 511.0
##  Mean   : -55.29   Mean   : -312.6   Mean   : 380.1   Mean   : 393.6
##  3rd Qu.:  26.00   3rd Qu.:  -73.0   3rd Qu.: 737.0   3rd Qu.: 653.0
##  Max.   : 291.00   Max.   :  672.0   Max.   :1480.0   Max.   :1090.0
##  classe
##  A:5580
##  B:3797
##  C:3422
##  D:3216
##  E:3607
##
```

Since we will not use regression, highly-correlated feature pairs are not a problem. We assume any outliers are true measurements, since we don't have the means to check if they are indeed so. For the two categorical features: user_name and classe, we see from the output of summary(data1) above that there is no problem with imbalance. We will deal with skewed variables by applying the BoxCox preprocessing option from the caret package when using linear models.

We plot the outcome variable in the plane of the first two principal components.

```
library(stats)
pr.out = prcomp(data1[,2:53], scale=TRUE)
plot(pr.out$x[,1],pr.out$x[,2],col=data1$classe,xlab = "First Principal Component", ylab="Second Princi
```



We see from the plot that the classes are not easily separable and that principal component analysis is probably of little use here.

Feature Preprocessing/Selection/Extraction

There are no missing data as we saw above.

We now scale the data to have mean 0 and SD 1.

```
data1[, 2:53] <- as.data.frame(lapply(data1[, 2:53],scale))
```

We also create a second dataset which includes only the variables seen by single-factor ANOVA to have an association with the outcome variable.

```
relatedVariables <- which(sapply(names(data1[, 2:53]),
                                 function (x) { anova(lm(
                                     as.formula(paste(x, " ~ classe")),
                                     data = data1))[["Pr(>F)"]][1]}) < 0.05) + 1
data2 <- data1[, c(1, relatedVariables, 54)]
```

Statistical prediction/modeling

The predictive models we will consider are : LDA, random forests and SVM.

We create a trainset and a model stacking validation set.

```
library(caret)
```

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

```
inTrain <- createDataPartition(y=data1$classe, p=0.7, list=FALSE)
training1 <- data1[inTrain, ]; testing1 <- data1[-inTrain, ]
training2 <- data2[inTrain, ]; testing2 <- data2[-inTrain, ]
```

We create 10 folds for k-fold cross-validation.

```
kNum <- 10
folds <- createFolds(training1$classe, k = kNum)
```

Linear Discriminant Analysis We train the LDA model on the data.

```
cv_results <- sapply(folds, function(x) {
    data_train <- training1[x, ]
    data_test <- training1[-x, ]
    data_model <- train(classe ~ . , data = data_train,
                        method = "lda", preProcess = "BoxCox")
    data_pred <- predict(data_model, newdata = data_test)
    return(mean(data_pred == data_test$classe))
})
```

```
## Loading required package: MASS
```

```
print(paste("With 10-fold CV, LDA accuracy has mean", round(mean(cv_results), 4),
            "and SD", round(sd(cv_results), 4),
            ". This is our estimate for out of sample accuracy."))
```

```
## [1] "With 10-fold CV, LDA accuracy has mean 0.7247 and SD 0.005 . This is our estimate for out of sa
```

Try again only with the relevant features.

```
cv_results <- sapply(folds, function(x) {
    data_train <- training2[x, ]
    data_test <- training2[-x, ]
    data_model <- train(classe ~ . , data = data_train,
                        method = "lda", preProcess = "BoxCox")
    data_pred <- predict(data_model, newdata = data_test)
    return(mean(data_pred == data_test$classe))
})
print(paste("With 10-fold CV, LDA accuracy has mean", round(mean(cv_results), 4),
            "and SD", round(sd(cv_results), 4),
            ". This is our estimate for out of sample accuracy."))
```

```
## [1] "With 10-fold CV, LDA accuracy has mean 0.725 and SD 0.0044 . This is our estimate for out of sa
```

Random Forests We train the random forests model on the data.

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
cv_results <- sapply(folds, function(x) {
    data_train <- training1[x, ]
    data_test <- training1[-x, ]
    data_model <- randomForest(classe ~ ., data = data_train, ntree = 500)
    data_pred <- predict(data_model, newdata = data_test)
    return(mean(data_pred == data_test$classe))
})
```

```
print(paste("With 10-fold CV, random forest accuracy has mean",
            round(mean(cv_results), 4),
            "and SD", round(sd(cv_results), 4),
            ". This is our estimate for out of sample accuracy."))
```

```
## [1] "With 10-fold CV, random forest accuracy has mean 0.9344 and SD 0.0046 . This is our estimate fo
```

Try again only with the relevant features.

```
cv_results <- sapply(folds, function(x) {
    data_train <- training2[x, ]
    data_test <- training2[-x, ]
    data_model <- randomForest(classe ~ ., data = data_train, ntree = 500)
    data_pred <- predict(data_model, newdata = data_test)
    return(mean(data_pred == data_test$classe))
})
```

```
print(paste("With 10-fold CV, random forest accuracy has mean",
            round(mean(cv_results), 4),
            "and SD", round(sd(cv_results), 4),
            ". This is our estimate for out of sample accuracy."))
```

```
## [1] "With 10-fold CV, random forest accuracy has mean 0.9338 and SD 0.0046 . This is our estimate fo
```

Support Vector Machines

We train the support vector machine model on the data.

```
library(e1071)
cv_results <- sapply(folds, function(x) {
    data_train <- training1[x, ]
    data_test <- training1[-x, ]
    data_model <- svm(classe ~ . , data = data_train, cost = 20)
    data_pred <- predict(data_model, newdata = data_test)
    return(mean(data_pred == data_test$classe))
})
```

```
print(paste("With 10-fold CV, SVM accuracy has mean",
            round(mean(cv_results), 4),
            "and SD", round(sd(cv_results), 4),
            ". This is our estimate for out of sample accuracy."))
```

```
## [1] "With 10-fold CV, SVM accuracy has mean 0.8963 and SD 0.0047 . This is our estimate for out of sa
```

Try again only with the relevant features.

```
cv_results <- sapply(folds, function(x) {
    data_train <- training2[x, ]
    data_test <- training2[-x, ]
    data_model <- svm(classe ~ . , data = data_train, cost = 20)
    data_pred <- predict(data_model, newdata = data_test)
    return(mean(data_pred == data_test$classe))
})
```

```
print(paste("With 10-fold CV, SVM accuracy has mean",
            round(mean(cv_results), 4),
            "and SD", round(sd(cv_results), 4),
            ". This is our estimate for out of sample accuracy."))
```

```
## [1] "With 10-fold CV, SVM accuracy has mean 0.8989 and SD 0.0054 . This is our estimate for out of sa
```

Conclusions

We see that removing irrelevant variables did not improve prediction performance. This is evidence for the ability of the models used to pick up the relevant features. We also see that model performance did not deteriorate upon removing the irrelevant variables. This is evidence for these variables being indeed irrelevant. The best model is random forests, which keeps all the variables. Based on 10-fold cross-validation, **we estimate the out of sample error of this model as 7%**. Per the instructions, we do not show here the prediction results for the 20 observations in the test set.