Practical Machine Learning Assignment

Cherylyn Ee

6 October 2016

Background

Devices such as Nike FuelBand, Fitbit 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, we will use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict the manner in which participants did the exercise. We will use the "classe" variable in the data to predict and test on 20 test cases available in the test data set.

Data Loading & Cleansing

The training data for this project is obtained from: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data is obtained from: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

```
traindat <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-trai
ning.csv")
testdat <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testi
ng.csv")
library(caret)
```

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

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

```
library(knitr)
library(randomForest)
```

```
## randomForest 4.6-12
```

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

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

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

We partition the training data (70% training, 30% validation) for cross validation purposes.

```
set.seed(131188)
inTrain <- createDataPartition(traindat$classe, p=0.7, list=FALSE)
training <- traindat[inTrain,]
testing <- traindat[-inTrain,]
str(training)
```

```
## 'data.frame':    13737 obs. of  160 variables:
##  $ X                       : int  3 5 6 7 8 9 11 12 14 15 ...
##  $ user_name               : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2 2
2 2 2 2 2 2 ...
##  $ raw_timestamp_part_1    : int  1323084231 1323084232 1323084232 1323084232 1
323084232 1323084232 1323084232 1323084232 1323084232 1323084232 ...
##  $ raw_timestamp_part_2    : int  820366 196328 304277 368296 440390 484323 500
302 528316 576390 604281 ...
##  $ cvtd_timestamp          : Factor w/ 20 levels "02/12/2011 13:32",..: 9 9 9 9
9 9 9 9 9 9 ...
##  $ new_window              : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1
...
##  $ num_window              : int  11 12 12 12 12 12 12 12 12 12 ...
##  $ roll_belt               : num  1.42 1.48 1.45 1.42 1.42 1.43 1.45 1.43 1.42
1.45 ...
##  $ pitch_belt              : num  8.07 8.07 8.06 8.09 8.13 8.16 8.18 8.18 8.21
8.2 ...
##  $ 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 ...
##  $ kurtosis_roll_belt      : Factor w/ 397 levels "","#DIV/0!","-0.016850",..:
1 1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_picth_belt     : Factor w/ 317 levels "","#DIV/0!","-0.021887",..:
1 1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_yaw_belt       : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1
1 ...
##  $ skewness_roll_belt      : Factor w/ 395 levels "","#DIV/0!","-0.003095",..:
1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_roll_belt.1    : Factor w/ 338 levels "","#DIV/0!","-0.005928",..:
1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_yaw_belt       : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1
1 ...
##  $ max_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_belt            : Factor w/ 68 levels "","#DIV/0!","-0.1",..: 1 1 1
1 1 1 1 1 1 1 ...
```

```
##  $ min_roll_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_belt           : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_belt             : Factor w/ 68 levels "","#DIV/0!","-0.1",..: 1 1 1
1 1 1 1 1 1 1 ...
##  $ amplitude_roll_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_belt     : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_belt       : Factor w/ 4 levels "","#DIV/0!","0.00",..: 1 1 1 1
1 1 1 1 1 1 ...
##  $ var_total_accel_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_belt        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_belt             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_belt             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_belt_x             : num  0 0.02 0.02 0.02 0.02 0.02 0.03 0.02 0.02 0 .
..
##  $ gyros_belt_y             : num  0 0.02 0 0 0 0 0 0 0 0 ...
##  $ gyros_belt_z             : num  -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.
02 -0.02 0 ...
##  $ accel_belt_x             : int  -20 -21 -21 -22 -22 -20 -21 -22 -22 -21 ...
##  $ accel_belt_y             : int  5 2 4 3 4 2 2 2 4 2 ...
##  $ accel_belt_z             : int  23 24 21 21 21 24 23 23 21 22 ...
##  $ magnet_belt_x            : int  -2 -6 0 -4 -2 1 -5 -2 -8 -1 ...
##  $ magnet_belt_y            : int  600 600 603 599 603 602 596 602 598 597 ...
##  $ magnet_belt_z            : int  -305 -302 -312 -311 -313 -312 -317 -319 -310
-310 ...
##  $ roll_arm                 : num  -128 -128 -128 -128 -128 -128 -128 -128 -128
-129 ...
##  $ pitch_arm                : num  22.5 22.1 22 21.9 21.8 21.7 21.5 21.5 21.4 21
.4 ...
##  $ 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 ...
##  $ var_accel_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_arm         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_arm              : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_arm              : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_arm_x              : num  0.02 0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 .
..
##  $ gyros_arm_y              : num  -0.02 -0.03 -0.03 -0.03 -0.02 -0.03 -0.03 -0.
03 0 0 ...
##  $ gyros_arm_z              : num  -0.02 0 0 0 0 -0.02 0 0 -0.03 -0.03 ...
##  $ accel_arm_x              : int  -289 -289 -289 -289 -289 -288 -290 -288 -288
-289 ...
```

```
##  $ accel_arm_y             : int  110 111 111 111 111 109 110 111 111 111 ...
##  $ accel_arm_z             : int  -126 -123 -122 -125 -124 -122 -123 -123 -124
-124 ...
##  $ magnet_arm_x            : int  -368 -374 -369 -373 -372 -369 -366 -363 -371
-374 ...
##  $ magnet_arm_y            : int  344 337 342 336 338 341 339 343 331 342 ...
##  $ magnet_arm_z            : int  513 506 513 509 510 518 509 520 523 510 ...
##  $ kurtosis_roll_arm       : Factor w/ 330 levels "","#DIV/0!","-0.02438",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_picth_arm      : Factor w/ 328 levels "","#DIV/0!","-0.00484",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_yaw_arm        : Factor w/ 395 levels "","#DIV/0!","-0.01548",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ skewness_roll_arm       : Factor w/ 331 levels "","#DIV/0!","-0.00051",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ skewness_pitch_arm      : Factor w/ 328 levels "","#DIV/0!","-0.00184",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ skewness_yaw_arm        : Factor w/ 395 levels "","#DIV/0!","-0.00311",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ max_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_arm       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ roll_dumbbell           : num  12.9 13.4 13.4 13.1 12.8 ...
##  $ pitch_dumbbell          : num  -70.3 -70.4 -70.8 -70.2 -70.3 ...
##  $ yaw_dumbbell            : num  -85.1 -84.9 -84.5 -85.1 -85.1 ...
##  $ kurtosis_roll_dumbbell  : Factor w/ 398 levels "","#DIV/0!","-0.0035",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_picth_dumbbell : Factor w/ 401 levels "","#DIV/0!","-0.0163",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_yaw_dumbbell   : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1
1 ...
##  $ skewness_roll_dumbbell  : Factor w/ 401 levels "","#DIV/0!","-0.0082",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ skewness_pitch_dumbbell : Factor w/ 402 levels "","#DIV/0!","-0.0053",..: 1
1 1 1 1 1 1 1 1 1 ...
##  $ skewness_yaw_dumbbell   : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1
1 ...
##  $ max_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_dumbbell        : Factor w/ 73 levels "","#DIV/0!","-0.1",..: 1 1 1
1 1 1 1 1 1 1 ...
##  $ min_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_dumbbell        : Factor w/ 73 levels "","#DIV/0!","-0.1",..: 1 1 1
1 1 1 1 1 1 1 ...
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

```
dim(training); dim(testing)
```

```
## [1] 13737   160
```

```
## [1] 5885  160
```

By looking at the training data, we can notice variables with too many NA values, low variance or have no relevancy to predict classe. We will remove these variables as predictors

```
# Remove too many NA variables
NAvar <- sapply(training, function(x) mean(is.na(x))) > 0.95
training <- training[, NAvar==FALSE]
testing <- testing[, NAvar==FALSE]
dim(training); dim(testing)
```

```
## [1] 13737    93
```

```
## [1] 5885   93
```

```
# Remove Nearly Zero Variance variables
zerovar <- nearZeroVar(training)
training <- training[, -zerovar]
testing <- testing[, -zerovar]
dim(training); dim(testing)
```

```
## [1] 13737    59
```

```
## [1] 5885   59
```

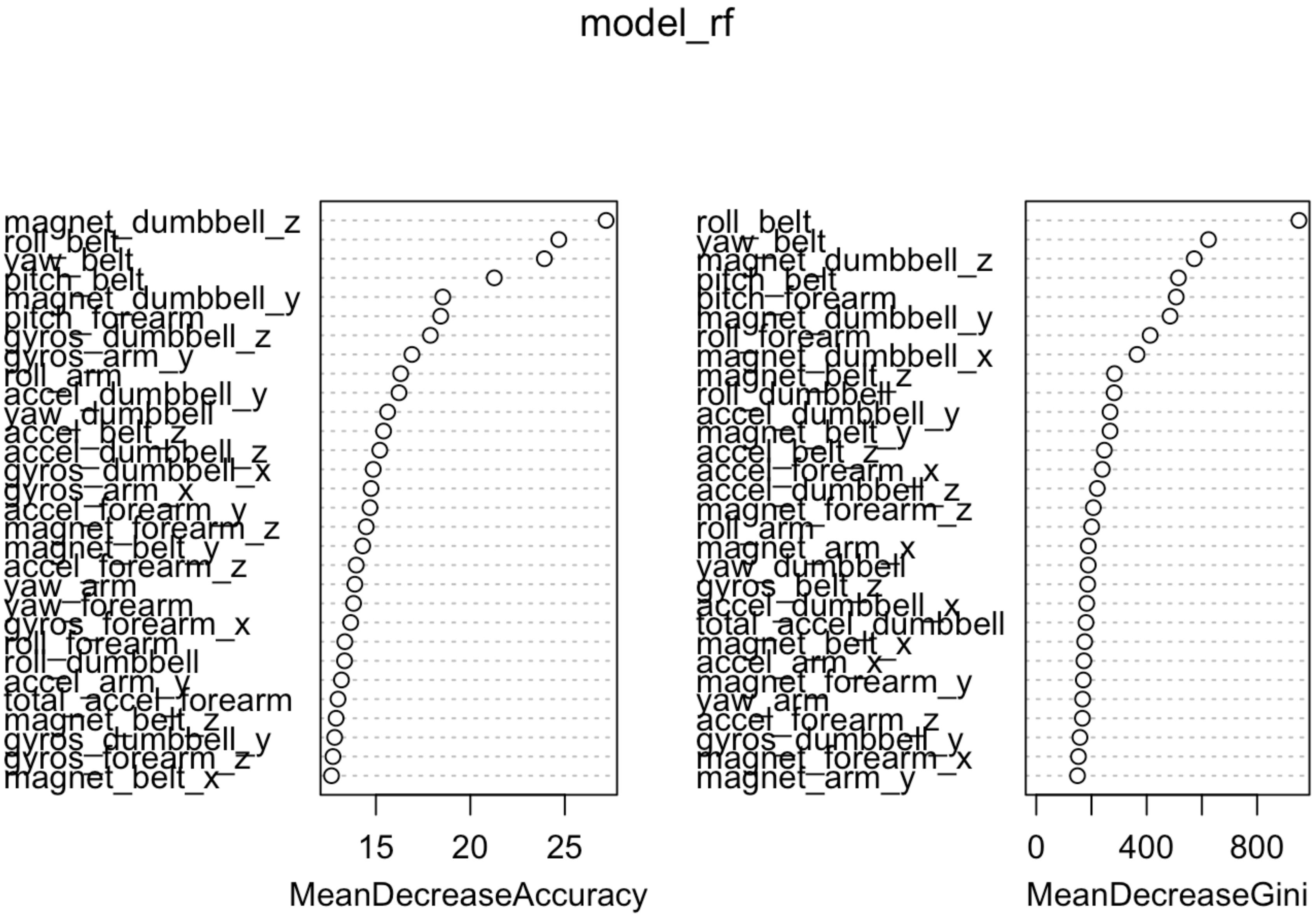
```
# Remove irrelevant variables
irrel <- grep("X|user_name|timestamp|num_window", names(training))
training <- training[,-irrel]
testing <- testing[, -irrel]
dim(training); dim(testing)
```

```
## [1] 13737    53
```

```
## [1] 5885   53
```

After data cleansing, number of predictors has been reduced to 53. However, this is still a large number of variables, thus, we will take a look at the importance of the predictors using Random Forest.

```
set.seed(131188)
model_rf <- randomForest(classe ~ ., data=training, ntree = 100, importance = TRUE
)
varImpPlot(model_rf, sort = TRUE)
```



From the we can choose the top 10 variables to simplify our model. Limiting the number of variables without sacrificing too much accuracy can ensure better interpretability of the model.

Ten top predictor: yaw_belt, row_belt, magnet_dumbbell_z, pitch_belt, pitch_forearm, magnet_dumbbell_y, roll_arm, roll_forearm, accel_dumbbell_y, accel_dumbbell_z

Check correlation between predictors

Generally, attributes with an absolute correlation of 0.75 or higher is removed.

```
correlationMatrix <- cor(training[,c("yaw_belt","roll_belt","accel_dumbbell_z","pi
tch_belt","magnet_dumbbell_z","magnet_dumbbell_y","pitch_forearm","accel_dumbbell_
y","roll_arm","roll_forearm")])
which(abs(correlationMatrix)>0.75, arr.ind=TRUE)
```

```
##                   row col
## yaw_belt            1   1
## roll_belt           2   1
## yaw_belt            1   2
## roll_belt           2   2
## accel_dumbbell_z    3   3
## pitch_belt          4   4
## magnet_dumbbell_z   5   5
## magnet_dumbbell_y   6   6
## pitch_forearm       7   7
## accel_dumbbell_y    8   8
## roll_arm            9   9
## roll_forearm       10  10
```

```
cor(training$yaw_belt, training$roll_belt)
```

```
## [1] 0.8156202
```

yaw_belt and roll_belt show high correlation with each other 0.8156202. We will remove yaw_belt from the variables to improve the model.

Model

We will use Random Forest method to model our training data with the 9 variables chosen.

```
set.seed(131188)
trctrl <- trainControl(method="cv", number = 3)
model_rf_2 <- train(classe ~., data = training, method = "rf", trControl=trctrl)
model_rf_2$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: 0.7%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3900    4    2    0    0 0.001536098
## B   20 2634    4    0    0 0.009029345
## C    0   18 2368   10    0 0.011686144
## D    0    0   22 2227    3 0.011101243
## E    0    1    4    8 2512 0.005148515
```

Prediction

```
predrf <- predict(model_rf_2, newdata = testing)
conMat <- confusionMatrix(predrf, testing$classe)
conMat
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1673   11    0    0    0
##          B    0 1125    5    1    0
##          C    0    3 1019    7    1
##          D    0    0    2  954    2
##          E    1    0    0    2 1079
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9941          
##                  95% CI : (0.9917, 0.9959)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9925          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9877   0.9932   0.9896   0.9972
## Specificity            0.9974   0.9987   0.9977   0.9992   0.9994
## Pos Pred Value         0.9935   0.9947   0.9893   0.9958   0.9972
## Neg Pred Value         0.9998   0.9971   0.9986   0.9980   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1912   0.1732   0.1621   0.1833
## Detection Prevalence   0.2862   0.1922   0.1750   0.1628   0.1839
## Balanced Accuracy      0.9984   0.9932   0.9955   0.9944   0.9983
```

```
predrf_full <- predict(model_rf, newdata = testing)
conMat_full <- confusionMatrix(predrf_full, testing$classe)
conMat_full
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    4    0    0    0
##          B    0 1134    4    0    0
##          C    0    1 1020    8    0
##          D    0    0    2  955    3
##          E    0    0    0    1 1079
##
## Overall Statistics
##
##                Accuracy : 0.9961
##                  95% CI : (0.9941, 0.9975)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9951
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9956   0.9942   0.9907   0.9972
## Specificity            0.9991   0.9992   0.9981   0.9990   0.9998
## Pos Pred Value         0.9976   0.9965   0.9913   0.9948   0.9991
## Neg Pred Value         1.0000   0.9989   0.9988   0.9982   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1927   0.1733   0.1623   0.1833
## Detection Prevalence   0.2851   0.1934   0.1749   0.1631   0.1835
## Balanced Accuracy      0.9995   0.9974   0.9961   0.9948   0.9985
```

Accuracy of predictions for the 9 variables is very close to using 53 variables (0.9941 ~ 0.9964).

Out of sample error rate

% out of sample error rate = 1 - Accuracy The out of sample error rate is approximately 0.6%

```
outofsampleerror <- 1 - sum(predrf == testing$classe)/length(predrf)
outofsampleerror
```

```
## [1] 0.005947324
```