Prediction Assignment Writeup

Samarjit Roy

July 1, 2016

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 for this Assignment

The training data 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.

Assignment Intended Results

The goal of our project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. We may use any of the other variables to predict with. We should create a report describing how we built our model, how we used cross validation, what we think the expected out of sample error is, and why we made the choices we did. We will also use our prediction model to predict 20 different test cases.

My submission for the Peer Review portion should consist of a link to a Github repo with my R markdown and compiled HTML file describing my analysis. The text of the writeup to < 2000 words and the number of figures to be less than 5.

Getting the data

```r
#Download Training Data
trainUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
trainDataset <- read.csv(url(trainUrl),header=T, na.strings=c("NA","#DIV/0!",""))
testDataset <- read.csv(url(testUrl),header=T, na.strings=c("NA","#DIV/0!",""))
```

Data Cleaning

Removing all variables with at least one "NA" were excluded from the analysis.

```
## NA exclusion for all available variables
dim(trainDataset)
```

```
## [1] 19622   160
```

```
nonNAtrainDS<-trainDataset[, apply(trainDataset, 2, function(x) !any(is.na(x)))]
nonNAtestDS<-testDataset[, apply(testDataset, 2, function(x) !any(is.na(x)))]
dim(nonNAtrainDS)
```

```
## [1] 19622    60
```

Removing Variables related to time and user information were excluded for a total of 53 variables and 19622 class measurements.

```
## variables with user information, time and undefined
cleanTrainDS<-nonNAtrainDS[,-c(1:6)]
cleanTestDS<-nonNAtestDS[,-c(1:6)]
dim(cleanTrainDS)
```

```
## [1] 19622    54
```

```
dim(cleanTestDS)
```

```
## [1] 20 54
```

Data Split

We randomly subsample 60% of the set for training purposes, while the 40% remainder will be used only for testing, evaluation and accuracy measurement.

```
inTrain <- createDataPartition(y=cleanTrainDS$classe, p=0.60, list=FALSE)
train1  <- cleanTrainDS[inTrain,]
train2  <- cleanTrainDS[-inTrain,]
dim(train1)
```

```
## [1] 11776    54
```

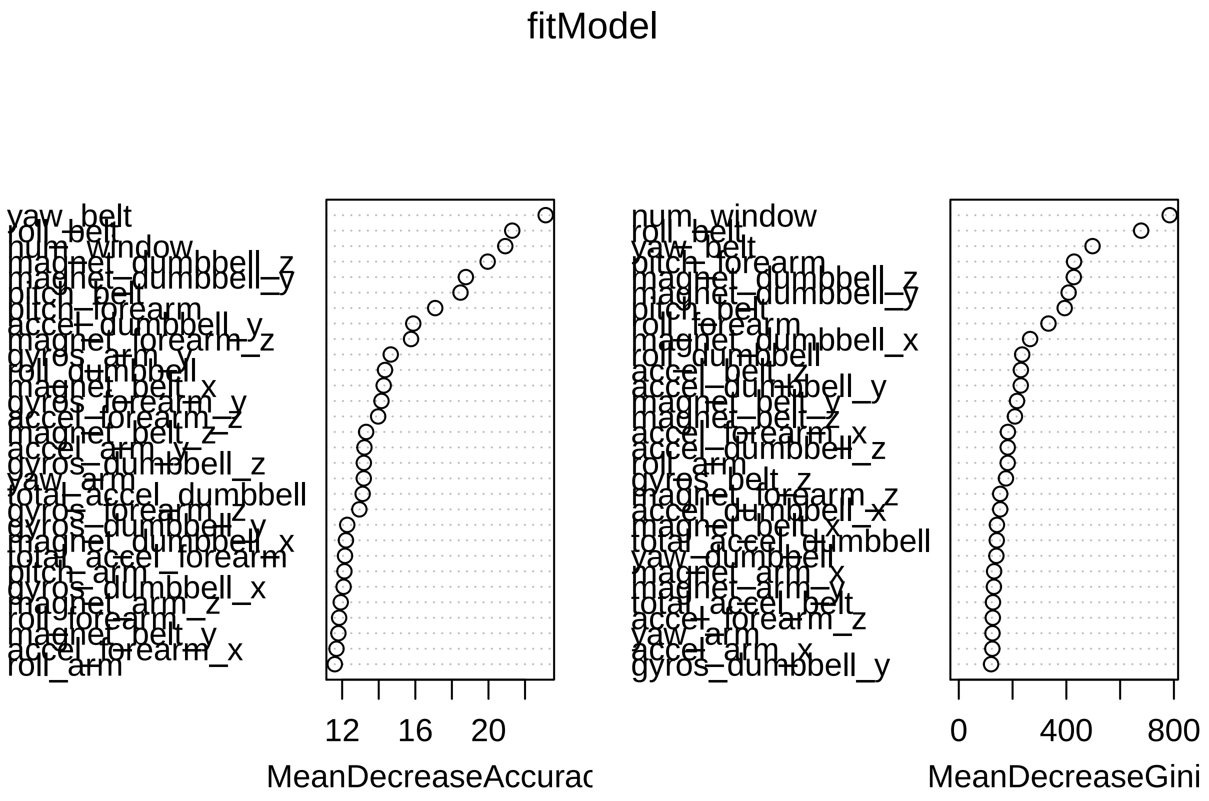
```
dim(train2)
```

```
## [1] 7846   54
```

Data Manipulation

53 covariates is a lot of variables. Let's look at their relative importance using the output of a quick Random Forest algorithm (which we call directly using randomForest() rather than the caret package purely for speed purposes as we cannot specify the number of trees to use in caret), and plotting data importance using varImpPlot():

```r
set.seed(3141592)
fitModel <- randomForest(classe~., data=train1, importance=TRUE, ntree=100)
varImpPlot(fitModel)
```



Using the Accuracy and Gini graphs above, we select the top 10 variables that we'll use for model building. If the accuracy of the resulting model is acceptable, limiting the number of variables is a good idea to ensure readability and interpretability of the model. A model with limited parameters (without unnessary elements) is certainly much more user friendly than a model with 53 parameters.

```r
impFitModelVar <- data.frame(fitModel$importance)
accNames<-row.names(impFitModelVar[order(-impFitModelVar$MeanDecreaseAccuracy),])[1:10]
giniNames<-row.names(impFitModelVar[order(-impFitModelVar$MeanDecreaseGini),])[1:10]
selectCovariates <- unique(append(accNames, giniNames))
```

Now Our 11 covariates are:

```r
selectCovariates
```

```
##  [1] "num_window"        "magnet_dumbbell_y" "roll_belt"
##  [4] "roll_forearm"      "yaw_belt"          "magnet_dumbbell_z"
##  [7] "pitch_belt"        "magnet_dumbbell_x" "pitch_forearm"
## [10] "roll_dumbbell"
```

Let's analyze the correlations between these 11 variables. The following code calculates the correlation matrix, replaces the 1s in the diagonal with 0s, and outputs which variables have an absolute value correlation above 75%:

```
correl = cor(train1[,selectCovariates])
diag(correl) <- 0
which(abs(correl)>0.75, arr.ind=TRUE)
```

```
##                   row col
## magnet_dumbbell_x   8   2
## yaw_belt            5   3
## roll_belt           3   5
## magnet_dumbbell_y   2   8
```

We may have a problem with roll__belt,yaw__belt,magnet__dumbbell__x,magnet__dumbbell__y which have a high correlation (above 75%) with each other. Let us check the correlation table:

```
cor(train1[,c("yaw_belt","roll_belt","magnet_dumbbell_x","magnet_dumbbell_y")])
```

```
##                      yaw_belt  roll_belt magnet_dumbbell_x
## yaw_belt           1.00000000  0.8132456       -0.02850823
## roll_belt          0.81324560  1.0000000        0.31359329
## magnet_dumbbell_x -0.02850823  0.3135933        1.00000000
## magnet_dumbbell_y -0.03304554 -0.2865846       -0.76794822
##                   magnet_dumbbell_y
## yaw_belt                -0.03304554
## roll_belt               -0.28658456
## magnet_dumbbell_x       -0.76794822
## magnet_dumbbell_y        1.00000000
```

We will remove "yaw__belt" and "magnet__dumbbell__x"and try the correlation test again.

```
selectCovariatesN <- selectCovariates [! selectCovariates %in% c("yaw_belt","magnet_dumbbell_x")]
selectCovariatesN
```

```
## [1] "num_window"        "magnet_dumbbell_y" "roll_belt"
## [4] "roll_forearm"      "magnet_dumbbell_z" "pitch_belt"
## [7] "pitch_forearm"     "roll_dumbbell"
```

```
correl = cor(train1[,selectCovariatesN])
diag(correl) <- 0
which(abs(correl)>0.75, arr.ind=TRUE)
```

```
##      row col
```

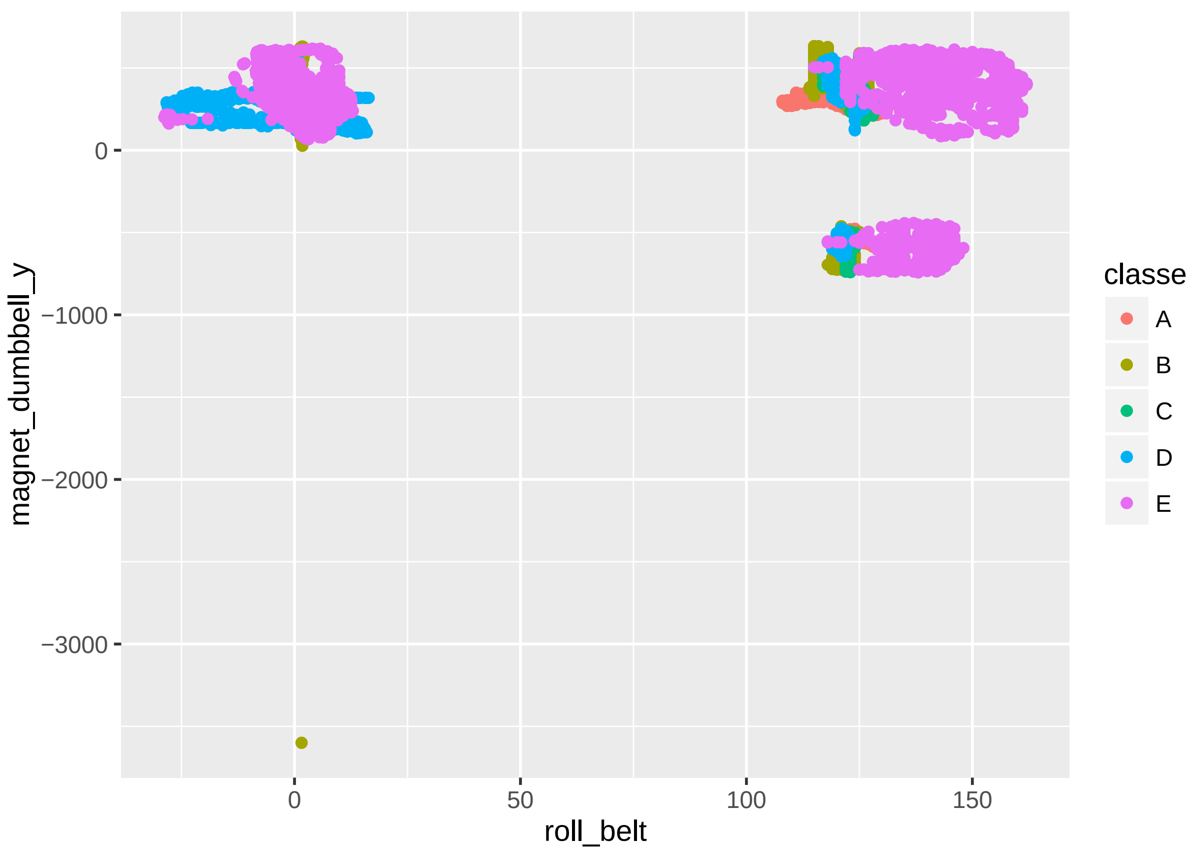
Two variables(roll__belt, magnet__dumbbell__y) are on top of the Accuracy and Gini graphs, and it may seem scary to eliminate one of them. Let's be bold and without doing any PCA analysis, we eliminate yaw__belt and magnet__dumbbell__x from the list of 10 variables and concentrate only on the remaining 8 variables.

Modeling

By re-running the correlation script above (eliminating yaw__belt) and outputting max(correl), we find that the maximum correlation among these 8 variables is 50.57% so we are satisfied with this choice of relatively independent set of covariates.

We can identify an interesting relationship between roll__belt and magnet__dumbbell__y:

```
qplot(roll_belt, magnet_dumbbell_y, colour=classe, data=train1)
```



This graph suggests that we could probably categorize the data into groups based on roll_belt values.

Incidentally, a quick tree classifier selects roll_belt as the first discriminant among all 53 covariates (which explains why we have eliminated yaw_belt instead of roll_belt, and not the opposite: it is a "more important" covariate):

```
fitModel <- randomForest(classe ~num_window+roll_belt+magnet_dumbbell_y+magnet_dumbbell_z+
                         roll_forearm+pitch_belt+pitch_forearm+accel_dumbbell_y , data=train1)
```

How accurate is this model?

We can use caret's confusionMatrix() function applied on train2 to get an idea of the accuracy:

```
prediction <- predict(fitModel, newdata=train2, type = "class")
confusionMat <- confusionMatrix(prediction, train2$classe)
confusionMat
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction     A    B    C    D    E
##            A 2231    1    0    0    0
##          B    1 1515    2    0    2
##          C    0    0 1366    4    0
##          D    0    2    0 1282    4
##          E    0    0    0    0 1436
## 
## Overall Statistics
##                                           
##                Accuracy : 0.998           
##                  95% CI : (0.9967, 0.9988)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9974          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9996   0.9980   0.9985   0.9969   0.9958
## Specificity            0.9998   0.9992   0.9994   0.9991   1.0000
## Pos Pred Value         0.9996   0.9967   0.9971   0.9953   1.0000
## Neg Pred Value         0.9998   0.9995   0.9997   0.9994   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1931   0.1741   0.1634   0.1830
## Detection Prevalence   0.2845   0.1937   0.1746   0.1642   0.1830
## Balanced Accuracy      0.9997   0.9986   0.9990   0.9980   0.9979
```

We have a very impressive accuracy (99.62%) which totally validates the idea / hypothesis we made to eliminate most variables and use only 8 relatively independent covariates.

```
sum(prediction != train2$classe) / length(train2$classe)
```

```
## [1] 0.002039256
```

Our out-of-sample error rate is 00.20%.

Predicting classe for pml-testing.csv

We predict the classification of the 20 observations of the testing data set for Coursera's "Course Project: Submission" challenge page:

```
testPrediction <- predict(fitModel, newdata=cleanTestDS)
cleanTestDS$classe <- testPrediction

testPrediction
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 
##  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
```