Predictive Modeling of Weight Lifting Excercises

INTRODUCTION

Fitness band devices track excercise activities of the wearers. A weight lifting project conducted by Velloso, Bulling, Gellersen, Ugulino and Fuks in 2013 evaluated the performance quality of the excercises executed by the device wearer. Their work focused on "the problem of specifying correct execution, the automatic and robust detection of execution mistakes, and how to provide feedback on the quality of execution to the user."

The data set used in this prediction project were collected from 6 young healthy male (20-28 years old) participants wearing accelerometers on their belt, forearm, arm and dumbell. All the participants had little weight lifting experience and replicated easily mistakes made in weight lifting exercises.

DATA SOURCE

The data from this project came from: http://groupware.les.inf.puc-rio.br/har. Both training and test data are provided by the course and downloaded from:

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

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

Whilet the training data has 19,622 rows and 160 columns, the test data has 20 rows and 160 columns. Based on the structure and summaries of the data, there were a number of column that are predominantly populated with "NA" and blanks. Some also have "#DIV/0!" entries. The following code was used to pre-process the data:

```
#  Read in data sets and define NA strings
# train = read.csv(file ="pml-training.csv",
header=TRUE,na.strings=c("NA","","#DIV/0!"))
# test = read.csv(file ="pml-testing.csv", header=TRUE, na.strings=c("NA",
""))

# Use str, describe and summary functions to check data for unique values
# Combine the two sets in preparation for data cleaning
# merge1 = train; merge2 = test
# merge1$type = "train"; merge2$type = "test"
# merge1$classe = NULL; merge2$problem_id = NULL
# merge = rbind(merge1, merge2)

# Delete columns that are predominantly "NA"
# mergeAll = merge[,complete.cases(t(merge))]

# Split the data back into training and testing sets
# training = subset(mergeAll, type == "train")
# testing = subset(mergeAll, type == "test")
```

```
# training$classe = train$classe
# training$type = NULL; testing$type = NULL
# testing$problem_id = test$problem_id
```

DATA SPLITTING

To do cross validation after the development of the predictive models, the training data set was split 60:40 for the train and test sets, respectively. There were 11,776 rows in the train set and 7,846 rows on the test set. Both have the number of predictors reduced to 60.

```
# Create training, cross-validation and testing sets for model building
# set.seed(12345)
# fitIndex = createDataPartition(training$classe, p = 0.60,list=FALSE)
# fitTrain = training[fitIndex,]
# fitTest = training[-fitIndex,]
```

MODEL BUILDING

The initial baseline accuracy was calculated in the test set and is shown below:

```
# Baseline Accuracy
# table(fitTest$classe)
#    A    B    C    D    E
# 2232 1518 1368 1286 1442
```

Random forest was chosen as the predictive model. Because random forest work by building a large collections of trees, (where each tree is split on a random subset of the available independent variable and built from a "bagged or bootstrapped" sample of the data), interpretability is hard. However,accuracy is increased compared to that of the predictive models of classification and regression trees.

The first model built included all the predictors and resulted into an accuracy of 100%, clearly a problem of overfitting. The second model was a function of all the predictors but excluding the variables X and user_name. These 2 variables are deemed unecessary as they were probably some sort of identifiers. The model's accuracy decreased to 0.998598. Again, the model predictions indicate overfitting. A third model was built with 3 additional predictors removed. Shown below are the code and the results of the third model which exhibited a slighly reduced accuracy of 0.9961764.

```
#
# set.seed(1)
# model2 = randomForest(classe~. -X -user_name -raw_timestamp_part_1 -
raw_timestamp_part_2
#                              -cvtd_timestamp-new_window, data = fitTrain)
# model2P = predict(model2, newdata=fitTest)

# table(fitTest$classe,model2P)
#   model2P
#       A    B    C    D    E
#  A 2232    0    0    0    0
#  B    4 1511    3    0    0
```

```
#   C     0     3 1364     1     0
#   D     0     0    14 1270     2
#   E     0     0     0     3 1439

# confusionMatrix(model2P, fitTest$classe)$overall["Accuracy"]
#   Accuracy
#   0.9961764
```

PREDICTION METRICS

Some of the metrics that can be used to help identify the important predictors are: 1) the number of times that a certain variable is selected for a split (Figure 1) and 2) the average reduction in impurity (Figure 2). Figure 1 shows that the following variables are the most important in terms of the number of splits: num_window, yaw_belt, pitch_belt,roll_belt, magnet_dumbell in x,y,z directions. Figure 2 shows that num_window is also the most important variable in terms of mean reduction in impurity followed by the other 5 shown in Figure 1 plus pitch_forearm. Figures (in both pdf and jpeg formats are available in https://github.com/myalopez/PracticalMachineLearning).

SUMMARY

The predictive model is able to accurately predict the type of excercise. This model uses 54 predictors and could probably use less if the predictive model is combined with another predictive model to identify the best combination of predictors.

REFERENCE

Velloso. E; Bulling, A.; Gellersen, H.; Ugulin W., W; Fuks, H. Qualitative Activity Recognition of Weight Lifting Excercises. Proceedings of 4th Interantional Conference in Cooperation with SIGCHI (Augmented Human '13), Stuttgart, Germany: ACM SIGCHI, 2013

Figure 1. Most important variables in terms of the number of splits.

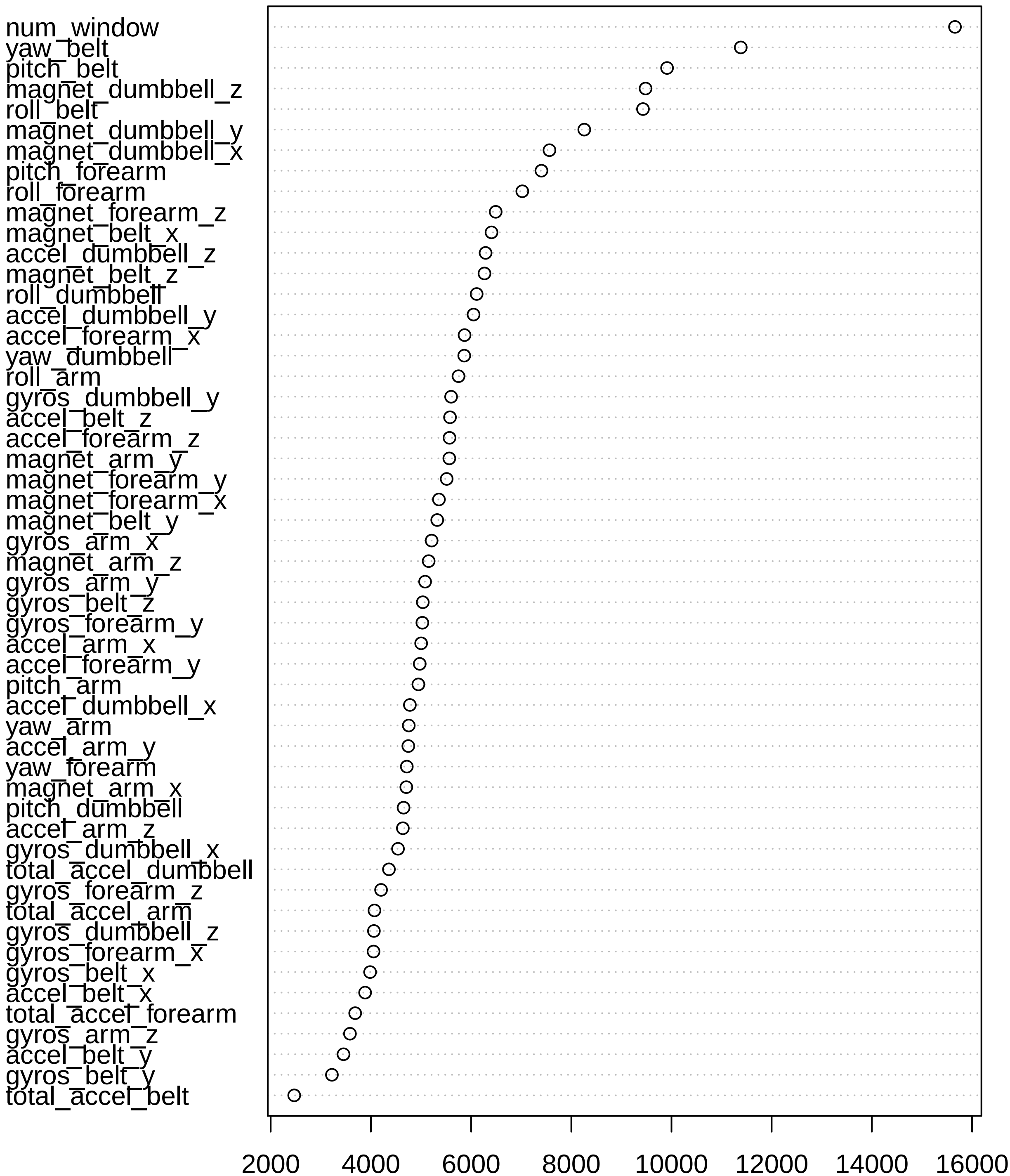

Figure 2. Most important variables in terms of mean reduction in impurity.