Coursera Course Practical Machine Learning Project Report

M. Liu

Contents

Data

The data for this project are from readings of wearable fitness trackers as descripted in the project instructiobns:

> "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 Preparation and Cleaning

The raw data are in two files. Since we will predict classes in pml-testing.csv dataset, we call this "validation" dataset instead.

```
train_in <- read.csv('./pml-training.csv', header=T)
validation <- read.csv('./pml-testing.csv', header=T)
```

Data Partitioning

We need to split the training data into training partition (70%) and testing partition (30%). We will use validation dataset (pml-testing.csv) for predicting classe.

```
set.seed(254)
training_sample <- createDataPartition(y=train_in$classe, p=0.7, list=FALSE)
training <- train_in[training_sample, ]
testing <- train_in[-training_sample, ]
```

Identification of Non-Zero Features in the Validation Dataset

In order to predict classes in the validation dataset, we need to use features that are non-zero in the dataset. We typically stay away from examining the data to avoid model fitting being influenced. Since this is not a time series analysis, looking at the data for non-zero data columns does not seem to be a major concern.

```
all_zero_colnames <- sapply(names(validation), function(x) all(is.na(validation[,x])==TRUE))
nznames <- names(all_zero_colnames)[all_zero_colnames==FALSE]
nznames <- nznames[-(1:7)]
nznames <- nznames[1:(length(nznames)-1)]
```

The non-zero features presented are:

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

Model Building

We use three (3) differnt models and compare their out-of-sample accuracty. The three models are:

1. Decision trees with CART (rpart)
2. Stochastic gradient boosting trees (gbm)
3. Random forest decision trees (rf)

The code to run fit these models is:

```
model_cart <- train(
  classe ~ .,
  data=training[, c('classe', nznames)],
  trControl=fitControl,
  method='rpart'
)
save(model_cart, file='./ModelFitCART.RData')
model_gbm <- train(
  classe ~ .,
  data=training[, c('classe', nznames)],
  trControl=fitControl,
  method='gbm'
)
save(model_gbm, file='./ModelFitGBM.RData')
model_rf <- train(
  classe ~ .,
  data=training[, c('classe', nznames)],
  trControl=fitControl,
  method='rf',
```

```
  ntree=100
)
save(model_rf, file='./ModelFitRF.RData')
```

Cross Validation

Cross validation is done for each model with K = 3. This is set in the above code chunk using the fitControl object as defined below:

```
fitControl <- trainControl(method='cv', number = 3)
```

Model Assessment (Out of sample error)

```
predCART <- predict(model_cart, newdata=testing)
cmCART <- confusionMatrix(predCART, testing$classe)
predGBM <- predict(model_gbm, newdata=testing)
cmGBM <- confusionMatrix(predGBM, testing$classe)
predRF <- predict(model_rf, newdata=testing)
cmRF <- confusionMatrix(predRF, testing$classe)
AccuracyResults <- data.frame(
  Model = c('CART', 'GBM', 'RF'),
  Accuracy = rbind(cmCART$overall[1], cmGBM$overall[1], cmRF$overall[1])
)
print(AccuracyResults)
```

```
##   Model  Accuracy
## 1  CART 0.4975361
## 2   GBM 0.9665251
## 3    RF 0.9974511
```

Based on an assessment of the out-of-sample errors, both gradient boosting and random forests outperform CART model, with random forests being slightly more accurate. The confusion matrix for the random forest model is below.

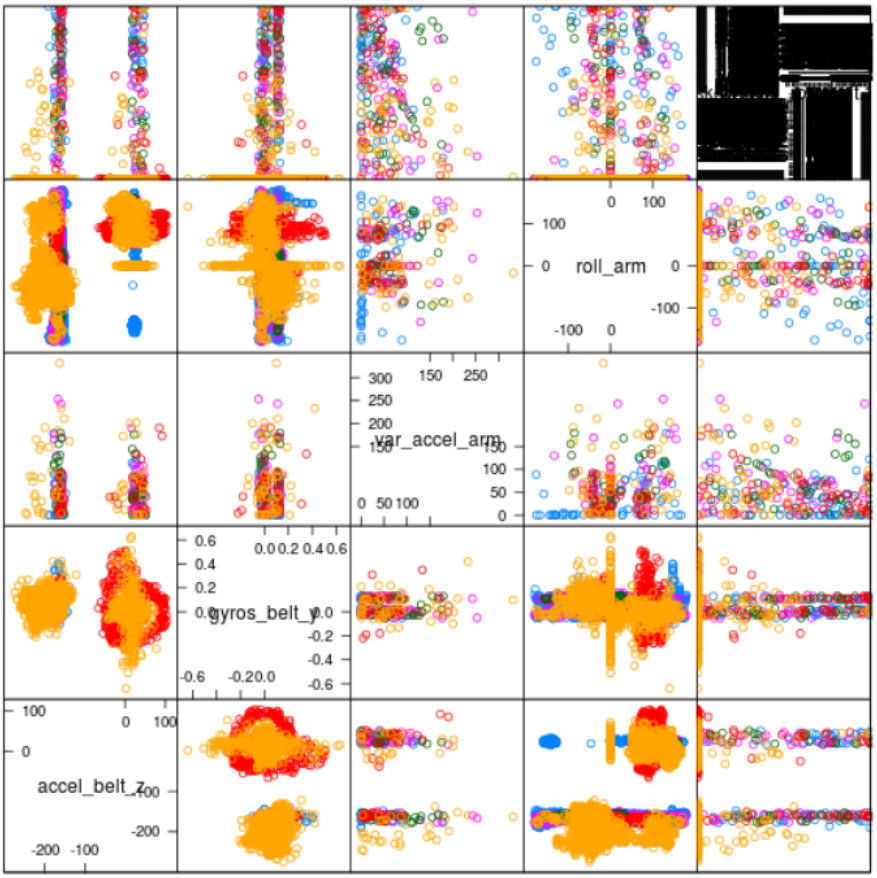
```
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    1    0    0    0
##          B    0 1137    2    0    0
##          C    0    1 1023    5    1
##          D    0    0    1  958    3
##          E    0    0    0    1 1078
```

The next step in modeling could be to create an ensemble model of these three model results, however, given the high accuracy of the random forest model, this process does not seem to be necessary. We will accept the random forest model as the champion and move on to predicting classe in the validation dataset (pml-testing.csv).

The champion model includes the following 5 features as the most important for predicting the exercise classe. A feature plot is included to show how these features are related to one another and how clusters of exercise classe begin to appear using these 5 features.

```
##        FeatureName Importance
## 1        roll_belt  100.00000
## 2       pitch_belt   59.40577
```

```
## 3         yaw_belt   54.09571
## 4 total_accel_belt   45.50496
## 5     gyros_belt_x   43.60051
```



Scatter Plot Matrix

Prediction

In the last step, we use the validation dataset ('pml-testing.csv') to predict a classe for each of the 20 observations based on the features contained in the dataset.

```
predValidation <- predict(champion_model, newdata=validation)
ValidationPredictionResults <- data.frame(
  problem_id=validation$problem_id,
  predicted=predValidation
```

```
)
print(ValidationPredictionResults)
```

```
##    problem_id predicted
## 1           1         B
## 2           2         A
## 3           3         B
## 4           4         A
## 5           5         A
## 6           6         E
## 7           7         D
## 8           8         B
## 9           9         A
## 10         10         A
## 11         11         B
## 12         12         C
## 13         13         B
## 14         14         A
## 15         15         E
## 16         16         E
## 17         17         A
## 18         18         B
## 19         19         B
## 20         20         B
```

Conclusion

Based on the data available, we fit a reasonably sound model with a high degree of accuracy in predicting out of sample observations. The random forest model is very accurate.

One constaint that could be relaxed in future work would be to remove the section of data preparation where we limit features to those that are non-zero in the validation dataset. We don't know Why there are so many missing features in the validation dataset, but not missing in the training dataset? Therefore, we are forced to apply the constaint. At leat for now and for this exercise, it seems necessary to apply the constaint. It is obvious and to to everyone's understanding that the features have to be ovelapped in training data and validation data to ensure that the model built is relevant.