Practical Machine Learning Final Report: Exercise Prediction

Data

The data for this project involves readings from wearable fitness trackers. The following is an excerpt from the Coursera project description:

> "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 (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset)".

Data Cleaning and Preparation

The raw data comes in to files, training and testing.

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

Data Partitioning

Since I'll be predicting classes in the testing dataset, I'll split the training data into training and testing partitions and use the pml-testing.csv as a validation sample. I'll use cross validation within the training partition to improve the model fit and then do an out-of-sample test with the testing partition.

```
set.seed(127)
training_sample <- createDataPartition(y=train_in$classe, p=0.7, list=FALSE)
training <- train_in[training_sample, ]
testing <- train_in[-training_sample, ]
```

Identification on Non-Zero Data

In order to predict classes in the validation sample, I'll need to use features that are non-zero in the validation data set. Typically, I'd stay away from the even looking at the validation data set so I'm not influenced by the contents in model fitting. However, since this is not a time series analysis, I feel that looking at the validation sample for non-zero data columns is not of major concern for finding a predictive model that fits well out of sample.

```
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 models will be fit using the following data columns:

```
##  [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

For this project I'll use 3 differnt model algorithms and then look to see whih provides the best out-of-sample accuracty. The three model types I'm going to test 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 0.4932880
## 2   GBM 0.9622770
## 3    RF 0.9926933
```

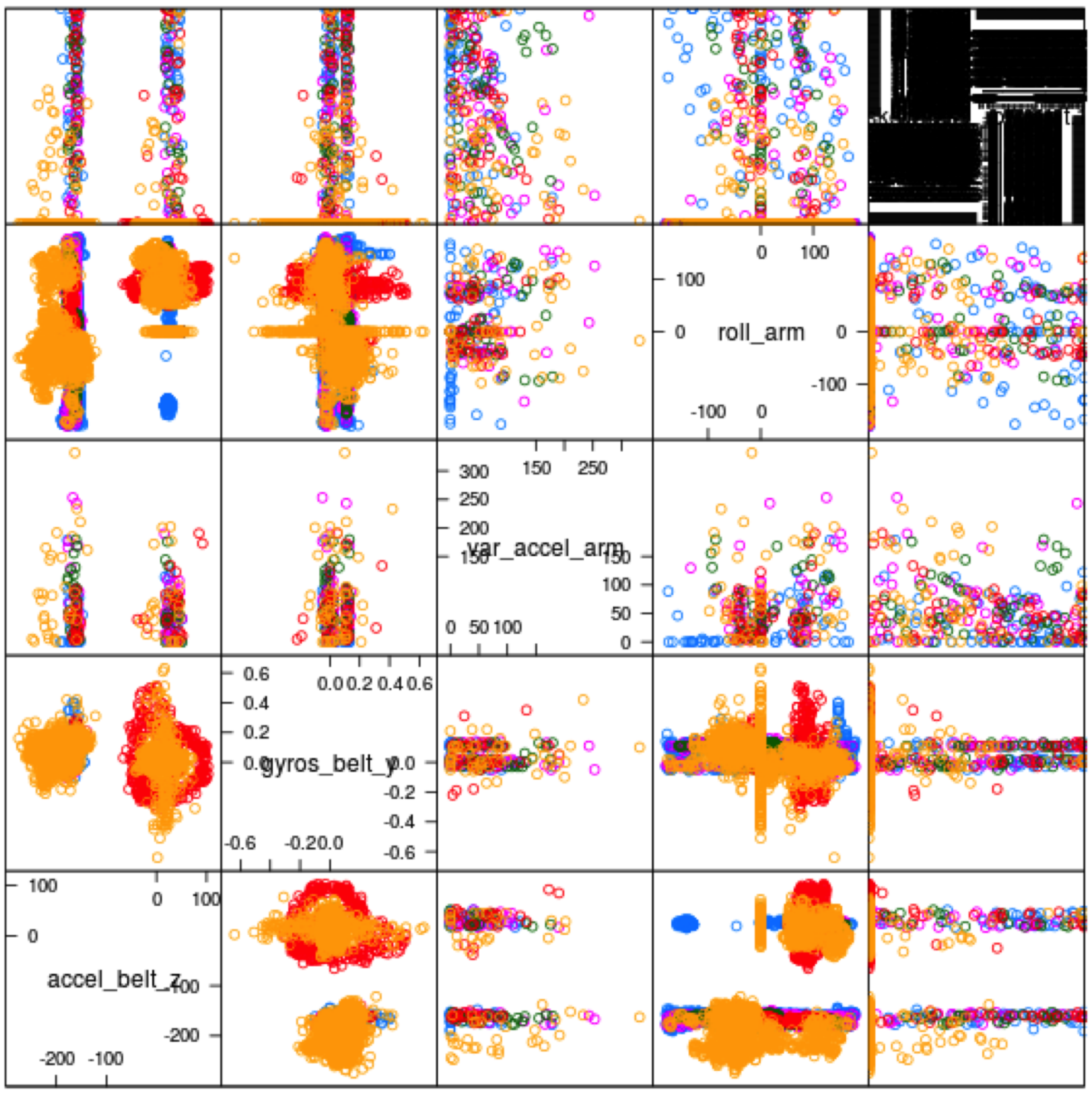
Based on an assessment of these 3 model fits and out-of-sample results, it looks like both gradient boosting and random forests outperform the CART model, with random forests being slightly more accurate. The confusion matrix for the random forest model is below.

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1671    9    0    0    0
##          B    3 1126    4    4    2
##          C    0    4 1020    6    1
##          D    0    0    2  952    6
##          E    0    0    0    2 1073
```

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, I don't believe this process is necessary here. I'll accept the random forest model as the champion and move on to prediction in the validation sample.

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

```
##        FeatureName  Importance
## 1        roll_belt  100.00000
## 2       pitch_belt   59.04099
## 3         yaw_belt   57.80767
## 4 total_accel_belt   44.25850
## 5     gyros_belt_x   42.48942
```



Scatter Plot Matrix

Prediction

As a last step in the project, I'll use the validation data sample ('pml-testing.csv') to predict a classe for each of the 20 observations based on the other information we know about these observations contained in the validation sample.

```
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, I am able to fit a reasonably sound model with a high degree of accuracy in predicting out of sample observations. One assumption that I used in this work that could be relaxed in future work would be to remove the section of data preparation where I limit features to those that are non-zero in the validation sample. For example, when fitting a model on all training data columns, some features that are all missing in the validation sample do included non-zero items in the training sample and are used in the decision tree models.

The question I'm left with is around the data collection process. Why are there so many features in the validation sample that are missing for all 20 observations, but these have observations in the training sample? Is this just introduced by the Coursera staff for the project to see how students respond? Or is it a genuine aspect of how data is collected from these wearable technologies?

Despite these remaining questions on missing data in the samples, the random forest model with cross-validation produces a surprisingly accurate model that is sufficient for predictive analytics.