Practical Machine Learning Project

- Overview
- Setup
- Exploratory Data Analysis
- Partitioning
- Training
- Approach 1: rpart
- Approach 2: rf with pca
- Final Assessment
- Summary
- Reference

Overview

In this report, Human Activity Recognition (HAR) motion dataset that is related five different exercises is imported into R program, and a prediction model will be build based on the subsetted training data. The HAR data contains following key variables with components:

- Accelration: x, y and z
- Gyro: x, y and z
- Magnet: x, y and z
- Tilt: yaw, pitch and roll

Additionally, there are other columns timestamp, person's name, variability are found in different columns.

Our goal is to estimate the correct outcome (classe) from those predictors. Analysis processes are presented below.

Setup

Before starting the analysis, some additional libraries are included. The main library is caret, and some graphing moudules are included to explore the dataset. I would like to note that parallel processing library allows us to register 4 cores of the CPU. This optional configuration that could be different per training environment, and it is mainly used in train() function in order to speed up the step.

```
library(caret)
```

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

```
library(AppliedPredictiveModeling)
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 3.3.0 Copyright (c) 2006-2014 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(ggplot2)
library(gridExtra)
```

```
## Loading required package: grid
```

```
library(doParallel)
```

```
## Loading required package: foreach
## Loading required package: iterators
## Loading required package: parallel
```

```
registerDoParallel(core=4)
```

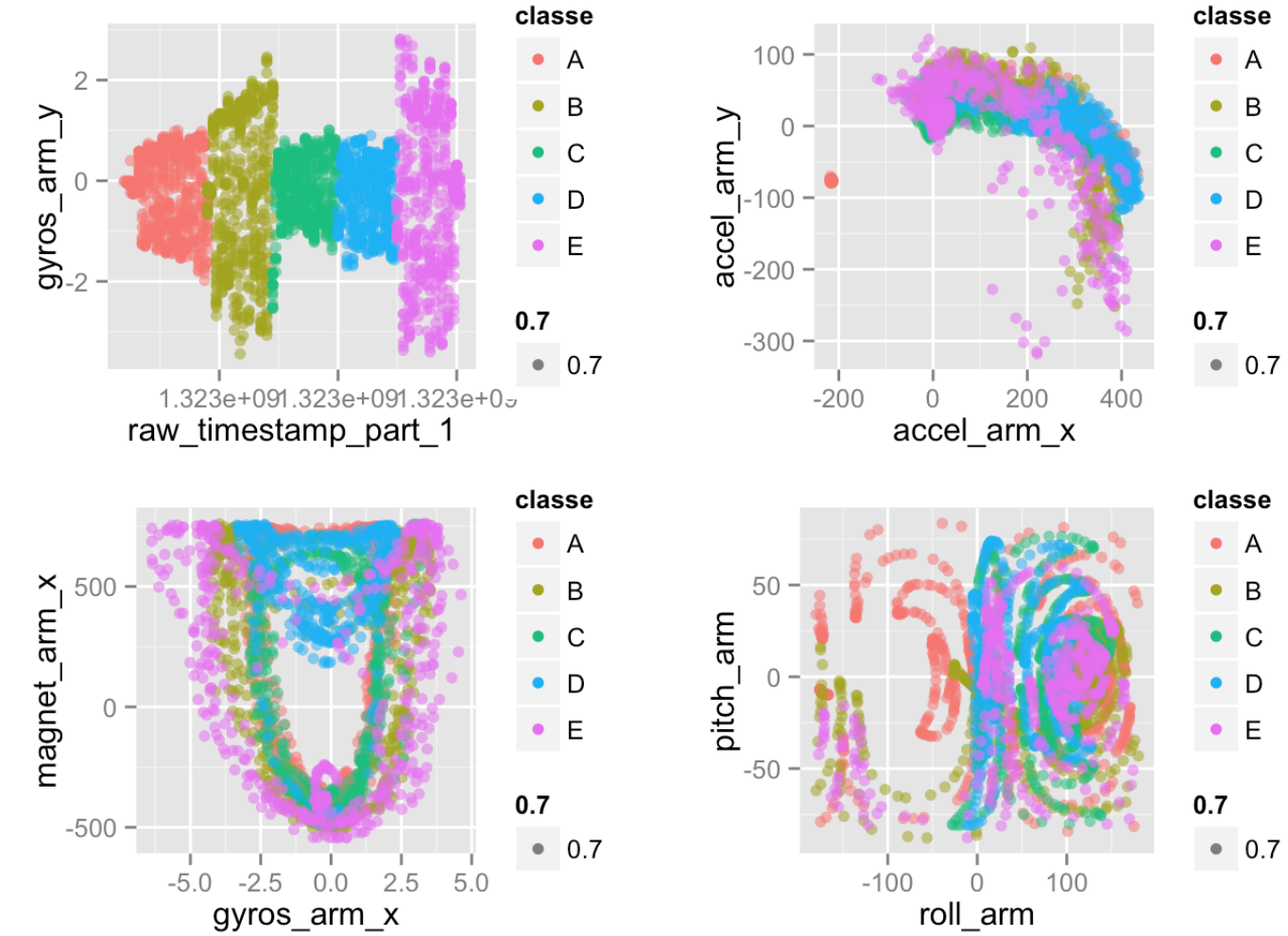
Two files will be downloaded and they will be loaded into memory.

```
if (!file.exists("pml-training.csv")){
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", destf
ile = "pml-training.csv", method="curl")
}
if (!file.exists("pml-testing.csv")){
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", destfi
le = "pml-testing.csv", method="curl")
}
training_src<-read.csv("pml-training.csv")
testing_final<-read.csv("pml-testing.csv")
```

Exploratory Data Analysis

To begin exploration with this dataset, I would like to extract sample subset by the person, Pedro, as an example. The first figure simply render different colors per exercise for gyro_arm_y component based on the timeline. The rest of three figures are few sample combinations of variables: x v.s. y of accelerometer values (top-right), magnet v.s. gyros of the same x component (bottom-left), and pitch v.s. roll of the arm (bottom-right).

```
tmp<-training_src[training_src$user_name=='pedro',]
g1<-qplot(raw_timestamp_part_1, gyros_arm_y,col=classe,alpha=0.7,data=tmp)
g2<-qplot(accel_arm_x,accel_arm_y,col=classe,alpha=0.7,data=tmp)
g3<-qplot(gyros_arm_x,magnet_arm_x,col=classe,alpha=0.7,data=tmp)
g4<-qplot(roll_arm,pitch_arm,col=classe,alpha=0.7,data=tmp)
grid.arrange(g1,g2,g3,g4, ncol=2,nrow=2)
```



By looking over those graphs, each cluster of exercises is not clearly separated; however, some segments could be visually identified in different area. For example, the bottom-right figure, pitch v.s. roll, presents the exercise type A (red) spreads all over the area while the exercise type D (blue) values gather from the center to positive areas in terms of the roll axis.

You might notice some outliers in the accelerometer value of the exercise type A.

Partitioning

With caret's createDataPartition function, the training and testing sets are splited into 70% and 30% ratio. Note that the classe variable was targetd for this partitioning process.

```
inTrain = createDataPartition(training_src$classe, p = 0.7, list=F)
training = training_src[inTrain,]
testing = training_src[-inTrain,]
```

Key variables are selected with grep function. As result of variable selection process, only 52 predictors are extracted out of 160 variables. By adding the outcome, the total number of columns is 53.

```
cols<-grep("^pitch|^yaw|^roll|^gyros|^accel|^magnet|^total|^classe",names(training))
length(cols)
```

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

Training

In this training section, I would like to start with “rpart” which uses relatively simple classification tree building algorithm, and then the “rf” method with PCA will be applied in the next approach.

Since the nature of he dataset shows overwrapped values over different exercise type, I expect this might generate lower accuracy than later approach.

In both steps, the default bootstrap option are used for the train control.

Approach 1: rpart

Train the model with only selected variables. See the final model and tree diaglram for more details.

```
fit <- train(classe ~.,method="rpart",data=training[,cols])
```

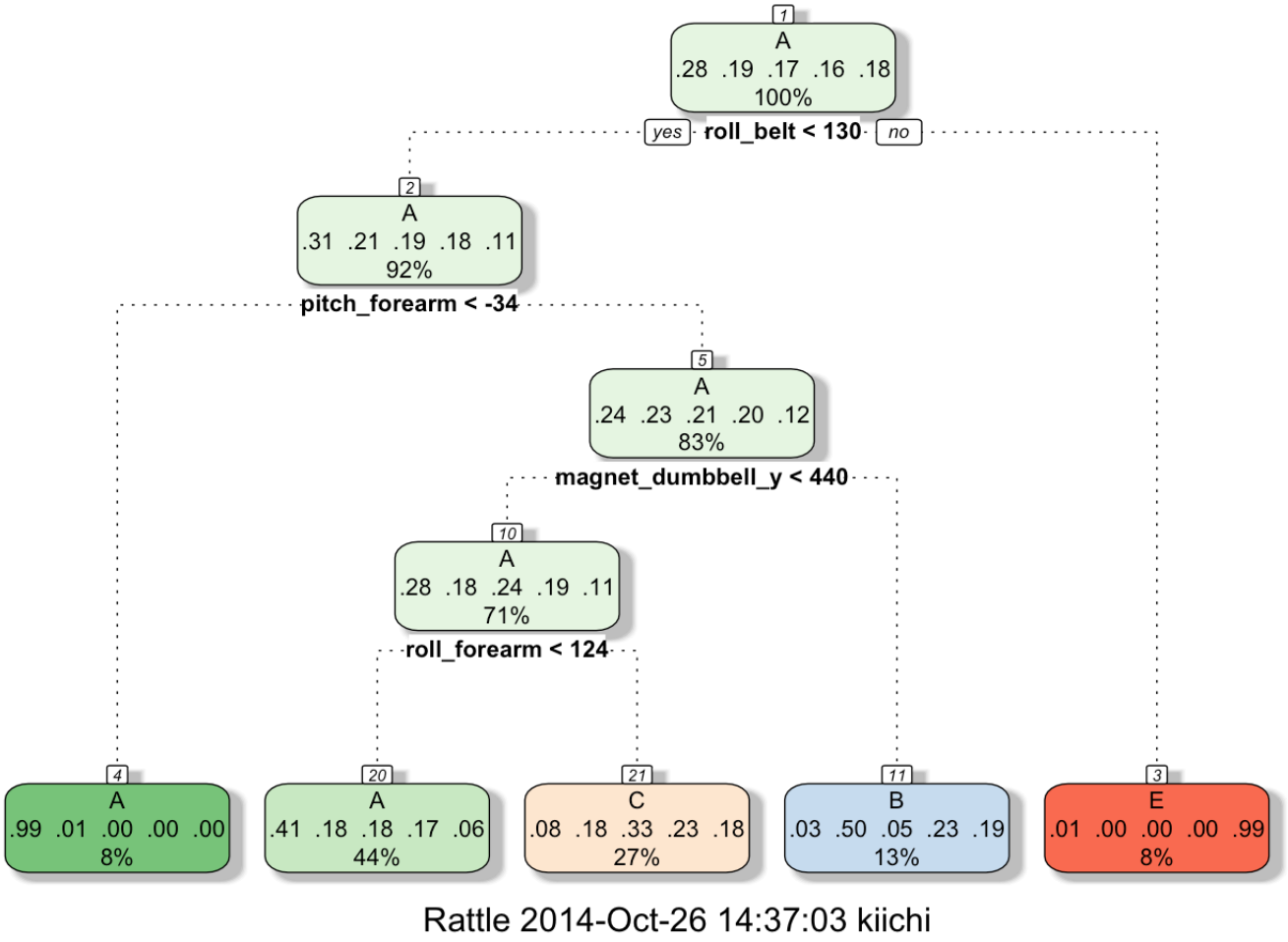
```
## Loading required package: rpart
```

```
fit$finalModel
```

```
## n= 13737
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 13737 9831 A (0.28 0.19 0.17 0.16 0.18)
##    2) roll_belt< 130.5 12571 8675 A (0.31 0.21 0.19 0.18 0.11)
##      4) pitch_forearm< -33.95 1105    7 A (0.99 0.0063 0 0 0) *
##      5) pitch_forearm>=-33.95 11466 8668 A (0.24 0.23 0.21 0.2 0.12)
##       10) magnet_dumbbell_y< 439.5 9692 6954 A (0.28 0.18 0.24 0.19 0.11)
##         20) roll_forearm< 124.5 6012 3569 A (0.41 0.18 0.18 0.17 0.061) *
##         21) roll_forearm>=124.5 3680 2465 C (0.08 0.18 0.33 0.23 0.18) *
##       11) magnet_dumbbell_y>=439.5 1774  884 B (0.034 0.5 0.046 0.23 0.19) *
##    3) roll_belt>=130.5 1166   10 E (0.0086 0 0 0 0.99) *
```

Here is the visualized tree structure.

```
fancyRpartPlot(fit$finalModel)
```



Using the partitioned testing set, the accuracy is displayed below.

```
est<-predict(fit,testing)
confusionMatrix(testing$classe,est)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1543   21  106    0    4
##          B  483  396  260    0    0
##          C  493   27  506    0    0
##          D  456  160  348    0    0
##          E  157  151  299    0  475
##
## Overall Statistics
##
##                Accuracy : 0.496
##                  95% CI : (0.483, 0.509)
##     No Information Rate : 0.532
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.34
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.493   0.5245    0.333       NA   0.9916
## Specificity             0.952   0.8552    0.881    0.836   0.8877
## Pos Pred Value          0.922   0.3477    0.493       NA   0.4390
## Neg Pred Value          0.623   0.9244    0.792       NA   0.9992
## Prevalence              0.532   0.1283    0.258    0.000   0.0814
## Detection Rate          0.262   0.0673    0.086    0.000   0.0807
## Detection Prevalence    0.284   0.1935    0.174    0.164   0.1839
## Balanced Accuracy       0.723   0.6898    0.607       NA   0.9397
```

The accuracy is 49.6 % with with Kappa is 0.34. The confusion matrix shows poor agreement to categorize all outcomes. Especially, D and E are dragging the entire accuracy. In this model, D is not identifiable at all.

Approach 2: rf with pca

Next, the Random Forest (rf) option will be selected as the method. PCA is applied for preprocessing the data. Note that the pca option will also do scaling and centering. See the final model below.

```
fit<-train(classe~.,method="rf",data=training[,cols],preProcess=c("pca"))
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
fit$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: 2
##
##         OOB estimate of  error rate: 2.59%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3868   16   15    5    2    0.009729
## B   52 2559   40    0    7    0.037246
## C    4   37 2329   22    4    0.027963
## D    6    1   93 2146    6    0.047069
## E    1   12   18   15 2479    0.018218
```

The final model includes 500 classification trees, and the estimates of error rate is about 14.21%.

Using the partitioned testing set, the accuracy is displayed below.

```
est<-predict(fit,testing)
confusionMatrix(testing$classe,est)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1661    3    8    1    1
##          B   21 1107   10    0    1
##          C    1   11 1004   10    0
##          D    0    1   43  919    1
##          E    1    4    3    9 1065
##
## Overall Statistics
##
##                Accuracy : 0.978
##                  95% CI : (0.974, 0.982)
##     No Information Rate : 0.286
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.972
##  Mcnemar's Test P-Value : 8.32e-08
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.986    0.983    0.940    0.979    0.997
## Specificity             0.997    0.993    0.995    0.991    0.996
## Pos Pred Value          0.992    0.972    0.979    0.953    0.984
## Neg Pred Value          0.995    0.996    0.987    0.996    0.999
## Prevalence              0.286    0.191    0.181    0.160    0.181
## Detection Rate          0.282    0.188    0.171    0.156    0.181
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       0.992    0.988    0.968    0.985    0.997
```

The confusion matrix for the testing set shows imporved accuracy, 97.8%, and the Kappa also increased to 0.972. The out of samples also shows resonable results in the table; There are some missing outcomes, however, it is not skewed result that was observed in above approach.

Final Assessment

Using given testing dataset that contains 20 rows, the model has been applied on them. The final classification results are shown below.

```
est<-predict(fit,testing_final[,cols])
est
```

```
##  [1] B A A A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

Summary

In conclusion, applying random method with PCA preprocessing option on the 70% partitioned training dataset achieved more than 97.8% of accuracy as the end. Assessment on the final testing set demonstrates 100% matching; none of outcome are appearing as out of samples.

Reference

Human Activity Recognition Project (http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har))