proj

JG

This is a classification problem, we selected 52 features based on device measurements, first used a simple tree classification method, which has high error rates. Then we used the random forest method, the out of bag (OOB) error rate is 0.0048, prediction error rate on the test set is 0.0054, very precise.

Import data:

```
training <- read.csv("training.csv", header = TRUE) # 19622 obs. of  160 variables:
```

Split the training data into: training/tesing by 0.7/0.3

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.1
```

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

```
set.seed(111)
rfTrain <- createDataPartition(y = training$classe, p = 0.7, list = FALSE)
rfTraining <- training[rfTrain, ]
rfTesting <- training[-rfTrain, ]
```

Based on the context, we select the features measures by the 4 devices (corresponding to 4 locations: belt arm, wist and dumbbell). Feature names start with: gyro, accel, magnet, roll, pitch, yaw, and total_accel. They are in total 52 features, together with the response variable, we have 53 variables in the selected data. Note that all 52 features are num and int type.

```
index53 <- c(grep('^gyro', names(rfTraining)),grep('^accel', names(rfTraining)),grep('^magnet', names(rfTraining)),gre
p('^roll',names(rfTraining)),grep('^pitch', names(rfTraining)),grep('^yaw', names(rfTraining)),grep('^total_accel',name
s(rfTraining)),160)
rfTraining53 <- rfTraining[, index53]
rfTesting53 <- rfTesting[, index53]
```

```
table(rfTraining53[, 53])
```

```
## 
##    A    B    C    D    E 
## 3906 2658 2396 2252 2525
```

Fit a simple classification tree

```
install.packages('tree')
library(tree)
treefit <- tree(classe ~., data = rfTraining53)
```

training error rate is: 0.3572. The following 11 Variables are used for spliting, which suggest that they might be important features in classifications.

```
[1] "roll_belt"            "pitch_forearm"        "roll_forearm"         "magnet_dumbbell_x"     "magnet_dumbbell_y"
[6] "magnet_dumbbell_z"    "roll_arm"             "yaw_belt"             "yaw_arm"              "total_accel_dumbbell"
[11] "accel_forearm_x
```

Predict with the fitted model on rfTesting

```
treepred <- predict(treefit, rfTesting53, type = "class")
with(rfTesting53, table(treepred, classe))
```

testing error rate is: 0.366

Now we try random forest method:

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.2.2
```

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

```
set.seed(121)
rffit <- randomForest(classe ~., data =  rfTraining53)
```

variables selected at every split = 7, number of trees = 500, out of bag (OOB) error rates: 0.48%, is very low. Note, in random forest method, there is no need to conduct cross-validation, because the OOB error rate plays the same role.

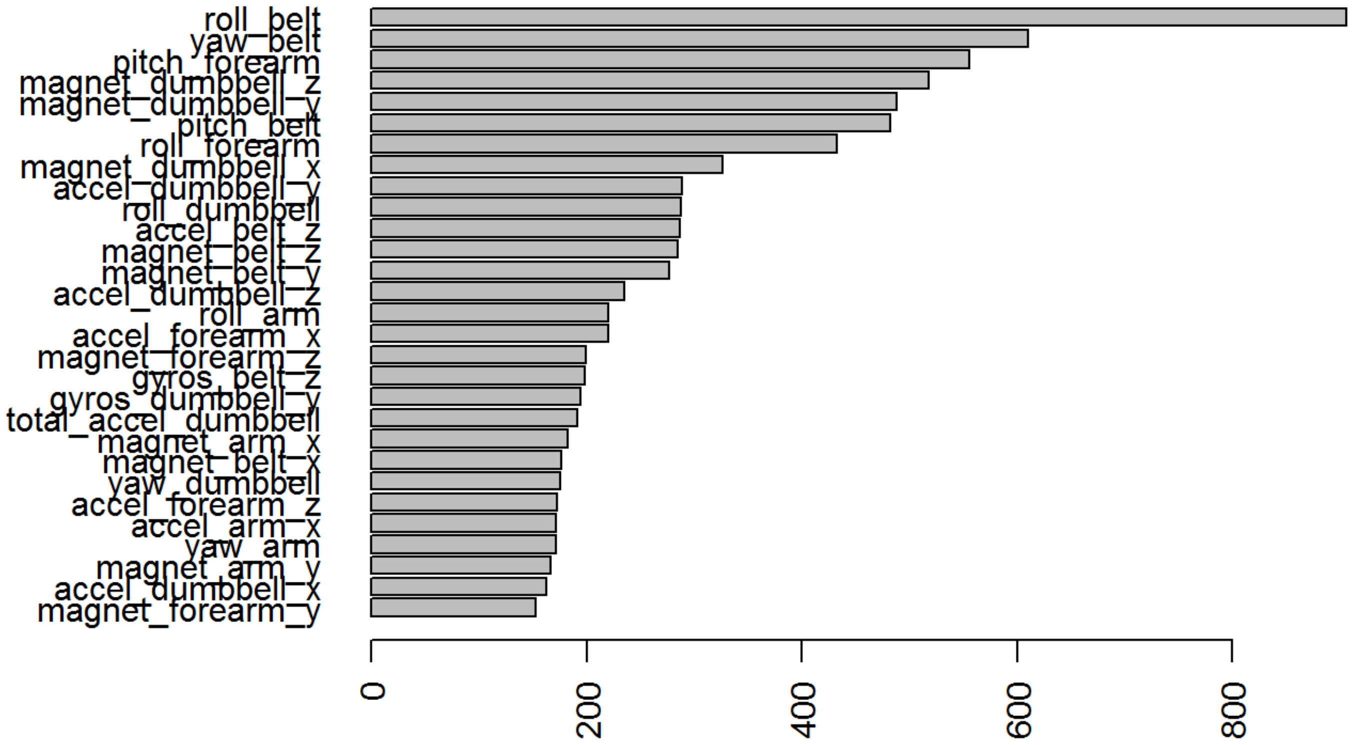
Predict with the fitted model on rfTesting.

```
rfpred <- predict(rffit, rfTesting53)
with(rfTesting53, table(rfpred, classe))
```

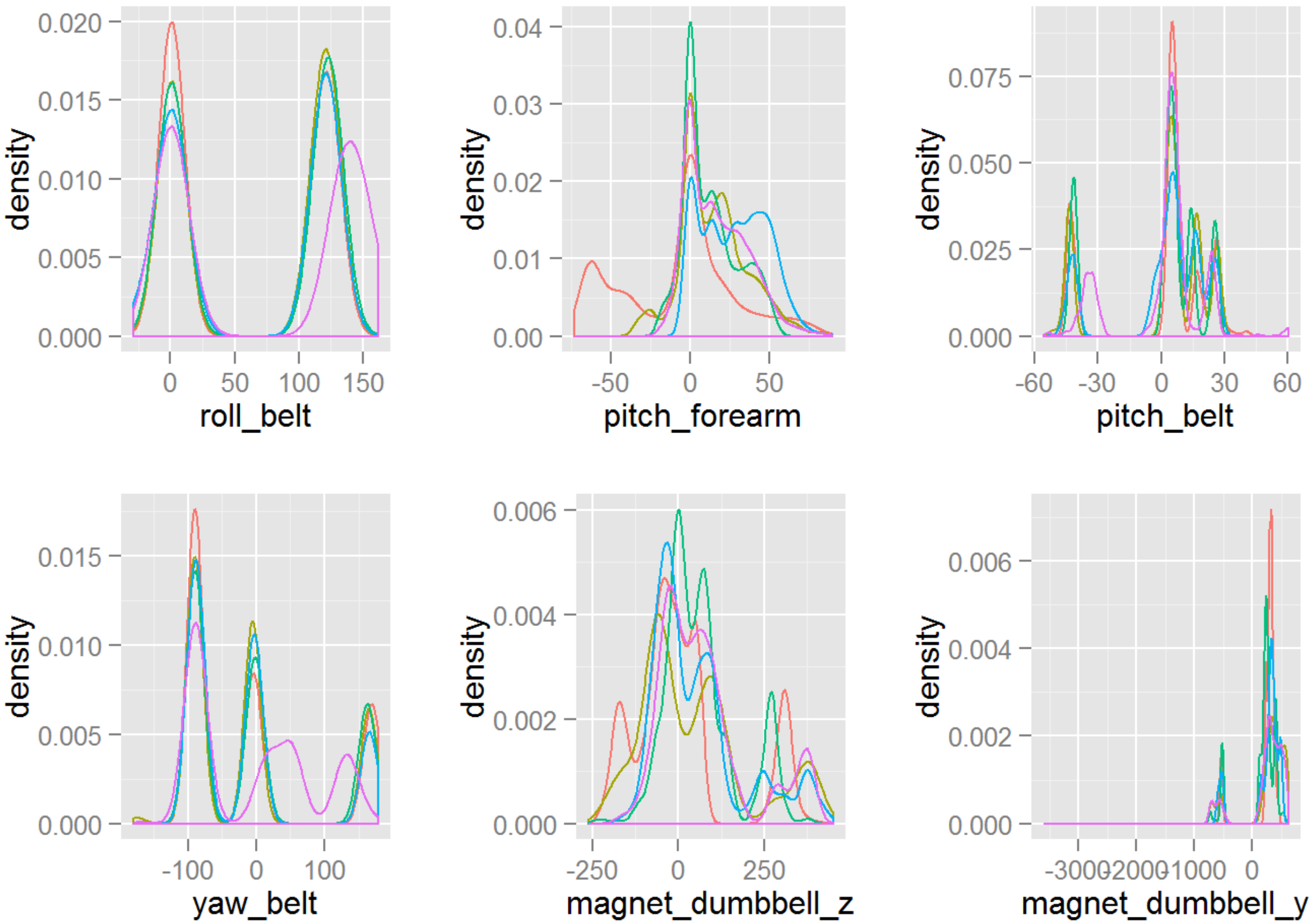
```
##       classe
## rfpred    A    B    C    D    E
##      A 1672    2    0    0    0
##      B    1 1126    3    0    0
##      C    0   11 1023   16    0
##      D    0    0    0  947    5
##      E    1    0    0    1 1077
```

The error rate is 0.54%, ie. the accurate rate is 99.46%

The following is a plot of the top 30 important variables in classification in the random forest process:



Plot of densities of the top 6 important variables: “roll_belt”, “yaw_belt”, “pitch_forearm”, “magnet_dumbbell_z”, “pitch_belt”, and “magnet_dumbbell_y”. Colored by variable “classe” (red stands for the correct way of excersize), we can see their classification abilities by the obvious difference between certain denisty curves.


density
0.000
0.005
0.010
0.015
0.020
0
50
100
150
roll_belt
density
0.00
0.01
0.02
0.03
0.04
-50
0
50
pitch_forearm
density
0.000
0.025
0.050
0.075
-60
-30
0
30
60
pitch_belt
density
0.000
0.005
0.010
0.015
-100
0
100
yaw_belt
density
0.000
0.002
0.004
0.006
-250
0
250
magnet_dumbbell_z
density
0.000
0.002
0.004
0.006
-3000
-2000
-1000
0
magnet_dumbbell_y