PML01

GgYy

20160710

1.Install needed packages

```
#install.packages("data.table")
#install.packages("caret")
#install.packages("randomForest")
#install.packages("foreach")
#install.packages("rpart")
#install.packages("rpart.plot")
#install.packages("corrplot")
```

2.Load needed packages

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.2.5
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.5
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.2.5
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(foreach)
```

```
## Warning: package 'foreach' was built under R version 3.2.5
```

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 3.2.5
```

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.2.5
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.2.5
```

3.Read datas

```
training_data <- read.csv("pml-training.csv", na.strings=c("#DIV/0!"," ",
"", "NA", "NAs", "NULL"))
testing_data <- read.csv("pml-testing.csv", na.strings=c("#DIV/0!"," ", "",
"NA", "NAs", "NULL"))
```

4.Clean datas Drop NAs, Drop highly corelated variables, drop variables whose contents are the same.

```
#4.1 Drop columns with NAs
str(training_data)
```

```
## 'data.frame':    19622 obs. of  160 variables:
##  $ X                       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ user_name               : Factor w/ 6 levels "adelmo","carlitos",..:
2 2 2 2 2 2 2 2 2 2 ...
##  $ raw_timestamp_part_1    : int  1323084231 1323084231 1323084231 132308
4232 1323084232 1323084232 1323084232 1323084232 1323084232 1323084232 ...
##  $ raw_timestamp_part_2    : int  788290 808298 820366 120339 196328 3042
77 368296 440390 484323 484434 ...
##  $ cvtd_timestamp          : Factor w/ 20 levels "02/12/2011 13:32",..:
9 9 9 9 9 9 9 9 9 9 ...
##  $ new_window              : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1
1 1 1 1 ...
##  $ num_window              : int  11 11 11 12 12 12 12 12 12 12 ...
##  $ roll_belt               : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.4
2 1.43 1.45 ...
##  $ pitch_belt              : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.1
3 8.16 8.17 ...
##  $ yaw_belt                : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -9
4.4 -94.4 -94.4 -94.4 ...
##  $ total_accel_belt        : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ kurtosis_roll_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_belt       : logi  NA NA NA NA NA NA ...
##  $ skewness_roll_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_belt.1    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_belt       : logi  NA NA NA NA NA NA ...
##  $ max_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_belt    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_belt      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_total_accel_belt    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_belt        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_belt       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_belt_x            : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02
0.03 ...
##  $ gyros_belt_y            : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z            : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.
02 -0.02 -0.02 0 ...
##  $ accel_belt_x            : int  -21 -22 -20 -22 -21 -21 -22 -22 -20 -2
1 ...
```

```
##  $ accel_belt_y            : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z            : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x           : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
##  $ magnet_belt_y           : int  599 608 600 604 600 603 599 603 602 60
9 ...
##  $ magnet_belt_z           : int  -313 -311 -305 -310 -302 -312 -311 -31
3 -312 -308 ...
##  $ roll_arm                : num  -128 -128 -128 -128 -128 -128 -128 -12
8 -128 -128 ...
##  $ pitch_arm               : num  22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 2
1.7 21.6 ...
##  $ yaw_arm                 : num  -161 -161 -161 -161 -161 -161 -161 -16
1 -161 -161 ...
##  $ total_accel_arm         : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ var_accel_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_arm         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_arm_x             : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.0
2 ...
##  $ gyros_arm_y             : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03
-0.02 -0.03 -0.03 ...
##  $ gyros_arm_z             : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02
-0.02 ...
##  $ accel_arm_x             : int  -288 -290 -289 -289 -289 -289 -289 -28
9 -288 -288 ...
##  $ accel_arm_y             : int  109 110 110 111 111 111 111 111 109 11
0 ...
##  $ accel_arm_z             : int  -123 -125 -126 -123 -123 -122 -125 -12
4 -122 -124 ...
##  $ magnet_arm_x            : int  -368 -369 -368 -372 -374 -369 -373 -37
2 -369 -376 ...
##  $ magnet_arm_y            : int  337 337 344 344 337 342 336 338 341 33
4 ...
##  $ magnet_arm_z            : int  516 513 513 512 506 513 509 510 518 51
6 ...
##  $ kurtosis_roll_arm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_arm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_pitch_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ min_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_arm       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ roll_dumbbell           : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell          : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell            : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ kurtosis_roll_dumbbell  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_picth_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ kurtosis_yaw_dumbbell   : logi  NA NA NA NA NA NA ...
##  $ skewness_roll_dumbbell  : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_pitch_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_yaw_dumbbell   : logi  NA NA NA NA NA NA ...
##  $ max_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

```
cleantraining <- training_data[, -which(names(training_data) %in% c("X", "us
er_name", "raw_timestamp_part_1", "raw_timestamp_part_2", "cvtd_timestamp",
"new_window", "num_window"))]
cleantraining = cleantraining[, colSums(is.na(cleantraining)) == 0]
#4.2 Drop variables with same content
zerovariance =nearZeroVar(cleantraining[sapply(cleantraining, is.numeric)],
saveMetrics=TRUE)
cleantraining = cleantraining[, zerovariance[, 'nzv'] == 0]
#4.3.1 Return the correlation matrix in matrix format
correlationmatrix <- cor(na.omit(cleantraining[sapply(cleantraining, is.nume
ric)]))
dim(correlationmatrix)
```

```
## [1] 52 52
```

```
correlationmatrixdegreesoffreedom <- expand.grid(row = 1:52, col = 1:52)
correlationmatrixdegreesoffreedom$correlation <- as.vector(correlationmatri
x)
#4.3.2 Remove highly correlated variables(up to 0.7)
removehighcorrelation <- findCorrelation(correlationmatrix, cutoff = .7, ver
bose = TRUE)
```

```
## Compare row 10  and column  1 with corr  0.992
##   Means:  0.27 vs 0.168 so flagging column 10
## Compare row 1  and column  9 with corr  0.925
##   Means:  0.25 vs 0.164 so flagging column 1
## Compare row 9  and column  22 with corr  0.722
##   Means:  0.233 vs 0.161 so flagging column 9
## Compare row 22  and column  4 with corr  0.759
##   Means:  0.224 vs 0.158 so flagging column 22
## Compare row 4  and column  3 with corr  0.762
##   Means:  0.2 vs 0.155 so flagging column 4
## Compare row 3  and column  8 with corr  0.708
##   Means:  0.2 vs 0.153 so flagging column 3
## Compare row 36  and column  29 with corr  0.849
##   Means:  0.257 vs 0.151 so flagging column 36
## Compare row 8  and column  2 with corr  0.966
##   Means:  0.229 vs 0.146 so flagging column 8
## Compare row 2  and column  11 with corr  0.884
##   Means:  0.212 vs 0.143 so flagging column 2
## Compare row 37  and column  38 with corr  0.769
##   Means:  0.198 vs 0.139 so flagging column 37
## Compare row 35  and column  30 with corr  0.773
##   Means:  0.195 vs 0.137 so flagging column 35
## Compare row 38  and column  5 with corr  0.781
##   Means:  0.177 vs 0.134 so flagging column 38
## Compare row 21  and column  24 with corr  0.814
##   Means:  0.176 vs 0.133 so flagging column 21
## Compare row 34  and column  28 with corr  0.808
##   Means:  0.176 vs 0.13 so flagging column 34
## Compare row 23  and column  26 with corr  0.779
##   Means:  0.137 vs 0.129 so flagging column 23
## Compare row 25  and column  24 with corr  0.792
##   Means:  0.145 vs 0.128 so flagging column 25
## Compare row 12  and column  13 with corr  0.779
##   Means:  0.122 vs 0.127 so flagging column 13
## Compare row 48  and column  51 with corr  0.772
##   Means:  0.145 vs 0.127 so flagging column 48
## Compare row 19  and column  18 with corr  0.918
##   Means:  0.095 vs 0.127 so flagging column 18
## Compare row 46  and column  45 with corr  0.846
##   Means:  0.131 vs 0.129 so flagging column 46
## Compare row 45  and column  31 with corr  0.71
##   Means:  0.098 vs 0.129 so flagging column 31
## Compare row 45  and column  33 with corr  0.716
##   Means:  0.078 vs 0.132 so flagging column 33
## All correlations <= 0.7
```

```
cleantraining <- cleantraining[, -removehighcorrelation]
#4.4 Generally drop blanks
for(i in c(8:ncol(cleantraining)-1)) {cleantraining[,i] = as.numeric(as.char
acter(cleantraining[,i]))}

for(i in c(8:ncol(testing_data)-1)) {testing_data[,i] = as.numeric(as.charac
ter(testing_data[,i]))}
#4.5 Redefine to be used data
featureset <- colnames(cleantraining[colSums(is.na(cleantraining)) == 0])
[-(1:7)]
modeldata <- cleantraining[featureset]
featureset
```

```
##  [1] "yaw_arm"              "total_accel_arm"      "gyros_arm_y"
##  [4] "gyros_arm_z"          "magnet_arm_x"         "magnet_arm_z"
##  [7] "roll_dumbbell"        "pitch_dumbbell"       "yaw_dumbbell"
## [10] "total_accel_dumbbell" "gyros_dumbbell_y"     "magnet_dumbbell_z"
## [13] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [16] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [19] "accel_forearm_x"      "accel_forearm_z"      "magnet_forearm_x"
## [22] "magnet_forearm_y"     "magnet_forearm_z"     "classe"
```

5.Build model Split 60% for training and 40% for testing.

```
idx <- createDataPartition(modeldata$classe, p=0.6, list=FALSE )
training <- modeldata[idx,]
testing <- modeldata[-idx,]
```

5 fold cross validation is used.

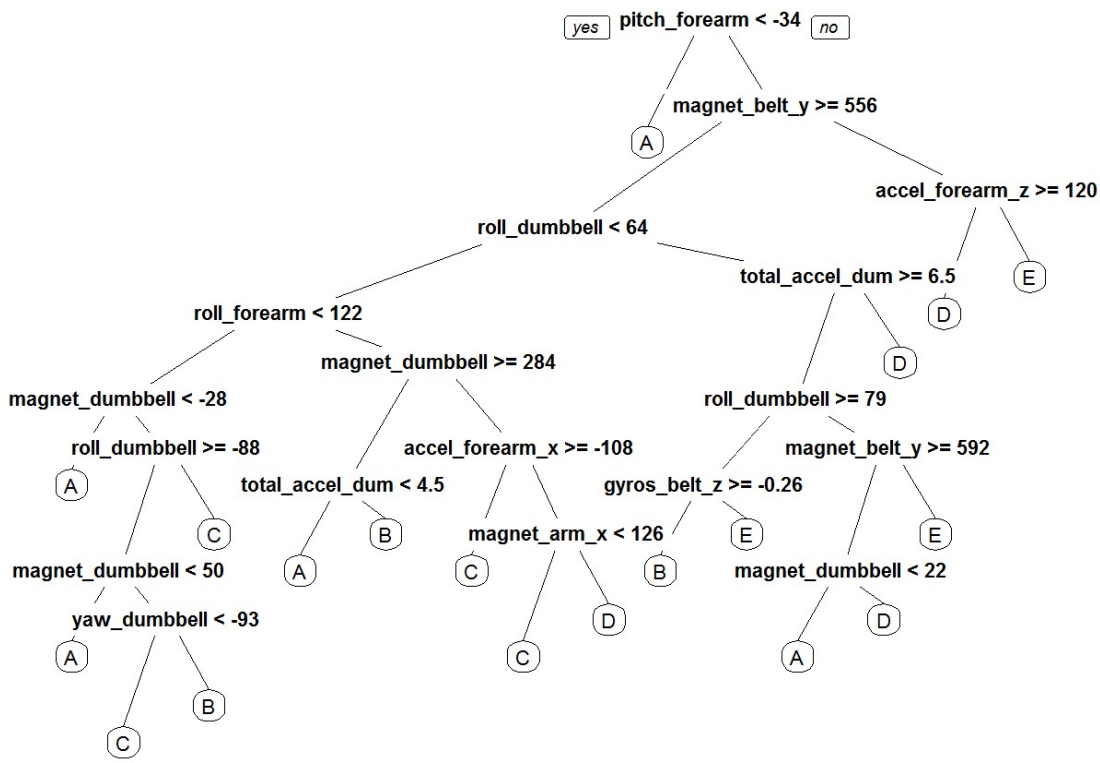
```
control <- trainControl(method="cv", 5)
model <- train(classe ~ ., data=training, method="rf", trControl=control, nt
ree=250)
model
```

```
## Random Forest
##
## 11776 samples
##    23 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 9420, 9422, 9420, 9422, 9420
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9719778  0.9645342
##   12    0.9680720  0.9595991
##   23    0.9609379  0.9505768
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

6.Predict Predict by using testing data

7.Draw Tress

```
treeModel <- rpart(classe ~ ., data=cleantraining, method="class")
prp(treeModel)
```



8.Answer the asked question

```
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FAL
SE)
  }
}

testing_data <- testing_data[featureset[featureset!='classe']]
answers <- predict(model, newdata=testing_data)
answers
```

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

```
pml_write_files(answers)
```