machine_learning_project

```
#Background
#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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).
#downloading and loading required R packages.
#summary
#The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(caret)
```

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

```
library(randomForest)
```

```
## randomForest 4.6-14
```

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

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

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

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

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

```
## corrplot 0.84 loaded
```

```
library(e1071)
#reading dataset
data.train<- read.csv("pml-training.csv", na.strings = c("NA", "#DIV/0!", ""))
data.test<- read.csv("pml-testing.csv", na.strings = c("NA", "#DIV/0!", ""))
dim(data.train)
```

```
## [1] 19622   160
```

```
#create a partition with the training dataset
data.train$cvtd_timestamp<- as.Date(data.train$cvtd_timestamp, format = "%m/%d/%Y %H:%M")
data.train$Day<-factor(weekdays(data.train$cvtd_timestamp))
table(data.train$classe)
```

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

```
#prediction on Test dataset
prop.table(table(data.train$classe))
```

```
##
##         A         B         C         D         E
## 0.2843747 0.1935073 0.1743961 0.1638977 0.1838243
```

```
prop.table(table(data.train$user_name))
```

```
##
##   adelmo  carlitos   charles    eurico    jeremy     pedro
## 0.1983488 0.1585975 0.1802059 0.1564570 0.1733768 0.1330140
```

```
prop.table(table(data.train$user_name,data.train$classe),1)
```

```
##
##                A         B         C         D         E
##   adelmo   0.2993320 0.1993834 0.1927030 0.1323227 0.1762590
##   carlitos 0.2679949 0.2217224 0.1584190 0.1561697 0.1956941
##   charles  0.2542421 0.2106900 0.1524321 0.1815611 0.2010747
##   eurico   0.2817590 0.1928339 0.1592834 0.1895765 0.1765472
##   jeremy   0.3459730 0.1437390 0.1916520 0.1534392 0.1651969
##   pedro    0.2452107 0.1934866 0.1911877 0.1796935 0.1904215
```

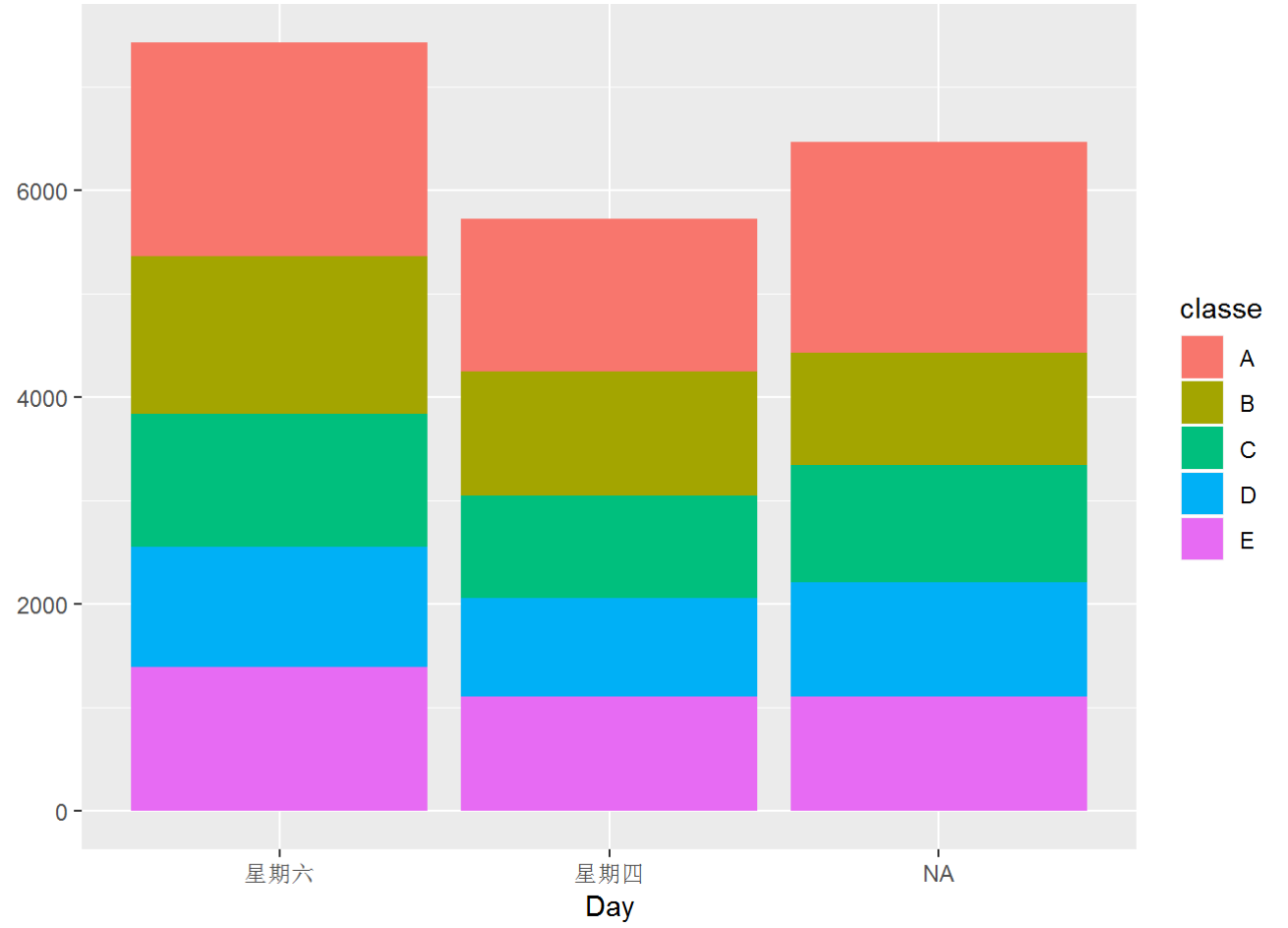
```
prop.table(table(data.train$user_name,data.train$classe),2)
```

```
##
##                A         B         C         D         E
##   adelmo   0.2087814 0.2043719 0.2191701 0.1601368 0.1901857
##   carlitos 0.1494624 0.1817224 0.1440678 0.1511194 0.1688384
##   charles  0.1611111 0.1962075 0.1575102 0.1996269 0.1971167
##   eurico   0.1550179 0.1559126 0.1428989 0.1809701 0.1502634
##   jeremy   0.2109319 0.1287859 0.1905319 0.1623134 0.1558082
##   pedro    0.1146953 0.1329997 0.1458212 0.1458333 0.1377876
```

```
prop.table(table(data.train$classe, data.train$Day),1)
```

```
##
##       星期六    星期四
##   A 0.5833804 0.4166196
##   B 0.5600147 0.4399853
##   C 0.5651030 0.4348970
##   D 0.5478220 0.4521780
##   E 0.5581302 0.4418698
```

```
#plot matrix results
qplot(x=Day, fill=classe, data = data.train)
```




```
data.train <- data.train[, colSums(is.na(data.train)) == 0]
data.test <- data.test[, colSums(is.na(data.test)) == 0]
classe<- data.train$classe
trainRemove<- grepl("^X|timestamp|window", names(data.train))
data.train<- data.train[, !trainRemove]
trainCleaned<- data.train[, sapply(data.train, is.numeric)]
trainCleaned$classe<- classe
testRemove<- grepl("^X|timestamp|window", names(data.test))
data.test<- data.test[, !testRemove]
testCleaned<- data.test[, sapply(data.test, is.numeric)]
set.seed(22519)
inTrain <- createDataPartition(trainCleaned$classe, p=0.70, list=F)
trainData <- trainCleaned[inTrain, ]
testData <- trainCleaned[-inTrain, ]
controlRf <- trainControl(method="cv", 5)
#Generalized Boosted Model
rfmod<- train(classe ~., data=trainData, method="rf", trControl=controlRf, importance=TRUE, ntree=100)
rfmod
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10989, 10991, 10988, 10989, 10991
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9902446  0.9876590
##   27    0.9911181  0.9887647
##   52    0.9852942  0.9813962
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

```
predictRfmod<- predict(rfmod, testData)
confusionMatrix(testData$classe, predictRfmod)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    A    B    C    D    E
##          A 1673    0    0    0    1
##          B    7 1128    4    0    0
##          C    0    0 1021    5    0
##          D    0    0   13  950    1
##          E    0    0    1    7 1074
##
## Overall Statistics
##
##               Accuracy : 0.9934
##                 95% CI : (0.991, 0.9953)
##    No Information Rate : 0.2855
##    P-Value [Acc > NIR] : < 2.2e-16
##
##                  Kappa : 0.9916
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9958   1.0000   0.9827   0.9875   0.9981
## Specificity            0.9998   0.9977   0.9990   0.9972   0.9983
## Pos Pred Value         0.9994   0.9903   0.9951   0.9855   0.9926
## Neg Pred Value         0.9983   1.0000   0.9963   0.9976   0.9996
## Prevalence             0.2855   0.1917   0.1766   0.1635   0.1828
## Detection Rate         0.2843   0.1917   0.1735   0.1614   0.1825
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9978   0.9988   0.9908   0.9923   0.9982
```

```
accuracy <- postResample(predictRfmod, testData$classe)
accuracy
```

```
##  Accuracy     Kappa
##  0.993373 0.991617
```

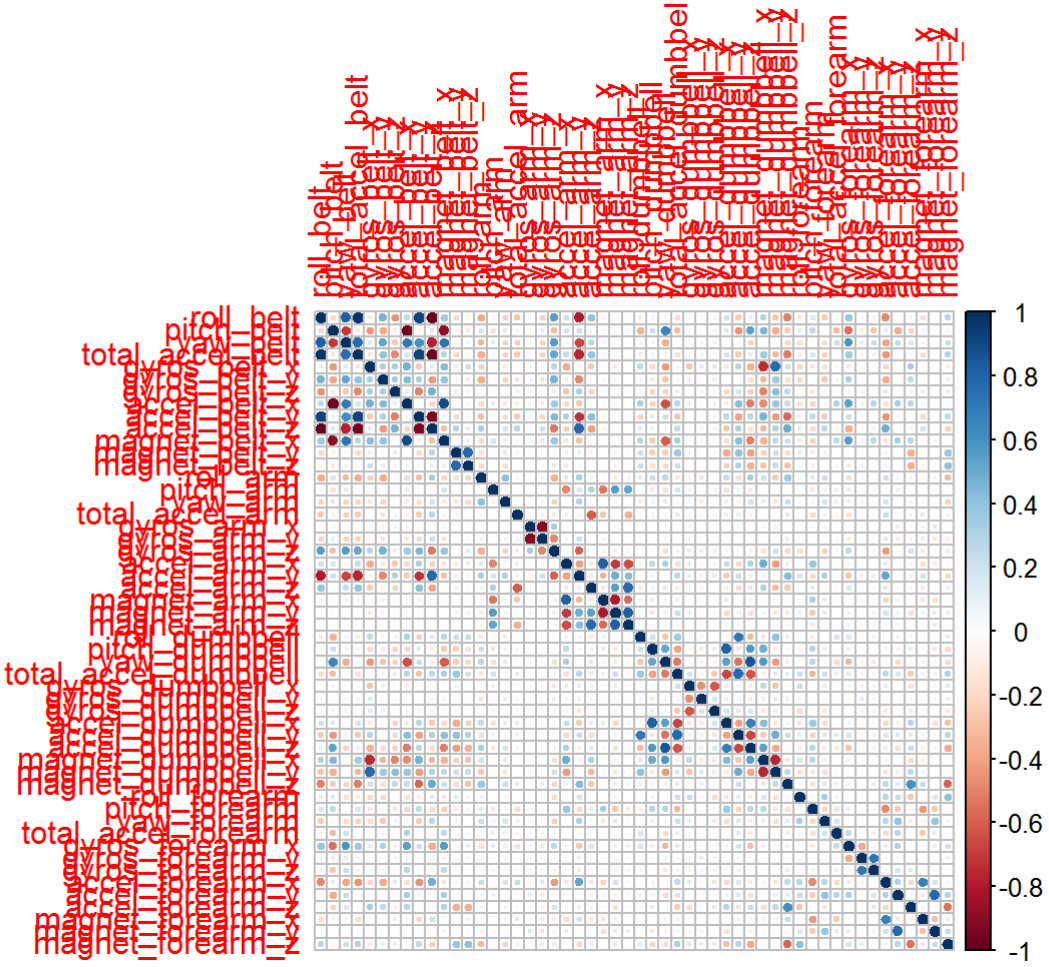
```
Error <- 1 - as.numeric(confusionMatrix(testData$classe, predictRfmod)$overall[1])
Error
```

```
## [1] 0.006627018
```

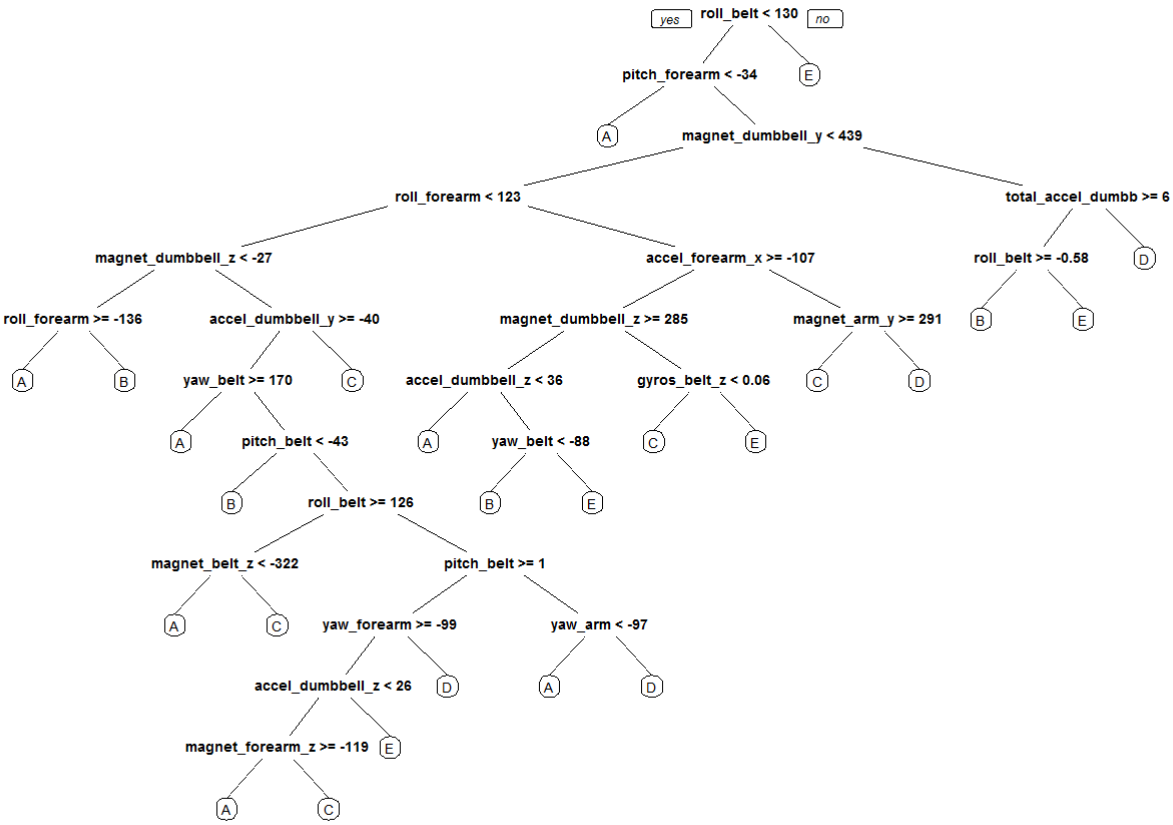
```
result <- predict(rfmod, testCleaned[, -length(names(testCleaned))])
result
```

```
##  [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
```

```
corrPlot <- cor(trainData[, -length(names(trainData))])
corrplot(corrPlot, method="circle")
```




```
#Applying the Selected Model to the Test Data.The accuracy of the 2 regression modeling methods above are: Random Forest : 0.9993 GBM : 0.9874 In that case, the Random Forest model will be applied to predict the quiz.
rtree<- rpart(classe ~ ., data=trainData, method="class")
prp(rtree)
```


yes
roll_belt < 130
no
pitch_forearm < -34
E
A
magnet_dumbbell_y < 439
roll_forearm < 123
total_accel_dumbb >= 6
magnet_dumbbell_z < -27
accel_forearm_x >= -107
roll_belt >= -0.58
D
roll_forearm >= -136
accel_dumbbell_y >= -40
magnet_dumbbell_z >= 285
magnet_arm_y >= 291
B
E
A
B
yaw_belt >= 170
C
accel_dumbbell_z < 36
gyros_belt_z < 0.06
C
D
A
pitch_belt < -43
A
yaw_belt < -88
C
E
B
roll_belt >= 126
B
E
magnet_belt_z < -322
pitch_belt >= 1
A
C
yaw_forearm >= -99
yaw_arm < -97
accel_dumbbell_z < 26
D
A
D
magnet_forearm_z >= -119
E
A
C