Machine Learning Project

JY.kim

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

1> Bring in the Data & Data claning

Other missing values were removed.

1-1#Get the Data and Claning

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

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

1-2# Delete missing variables

```
training <- training[, colSums(is.na(training))==0]

testing <- testing[, colSums(is.na(training))==0]
```

1-3# Delete 1-7 columns. it's not necessary data this project.

```
training <- training[, -c(1:7)]

testing <- testing[, -c(1:7)]
```

1-4# Delete 1-7 columns. it's not necessary data this project.

training <- training[, -c(1:7)]

testing <- testing[, -c(1:7)]

#overview data

View(training)

View(testing)

#inspecting about dataset

dim(training)

dim(testing)

```
> dim(training)
[1] 19622   53
> dim(testing)
[1]  20 153
```

2> #data modeling, using caret package

Library(caret)

intrain <- createDataPartition(y=training$classe, p = 0.7 ,list=F)

trainset <- training[intrain,]

testset <- training[-intrain,]

```
library(caret)
intrain <- createDataPartition(y=training$classe, p =
0.7 ,list=F)
trainset <- training[intrain,]
testset <- training[-intrain,]
```

First model > Decision Tree.

#modeling

library(rpart)

library(rattle)

mod <- rpart(classe ~. , data=trainset, method="class")

plot(mod)

fancyRpartPlot(mod)

#predict

pred1 <- predict(mod, testset, type="class")

confusionMatrix(pred1, testset$classe)

```
> confusionMatrix(pred1, testset$classe)
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1503  240   22   85   27
         B   39  582   46   27   64
         C   35  228  889   98  129
         D   68   64   68  675   60
         E   29   25    1   79  802

Overall Statistics

               Accuracy : 0.7563
                 95% CI : (0.7452, 0.7673)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6909
 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.8978   0.5110   0.8665   0.7002   0.7412
Specificity            0.9112   0.9629   0.8992   0.9472   0.9721
Pos Pred Value         0.8007   0.7678   0.6447   0.7219   0.8568
Neg Pred Value         0.9573   0.8914   0.9696   0.9416   0.9434
Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
Detection Rate         0.2554   0.0989   0.1511   0.1147   0.1363
Detection Prevalence   0.3189   0.1288   0.2343   0.1589
0.1590
Balanced Accuracy      0.9045   0.7369   0.8828   0.8237   0.8567
```

Decision tree's Accurancy overview 0.7563 and 95% CI(0.752~0.7673)

Model2. Random forest

####using Random Forest

library(randomForest)

mod2 <- randomForest(classe~., data=trainset, method="class")

pred2 <- predict(mod2, testset, type="class")

confusionMatrix(pred2, testset$classe)

```
> pred2 <- predict(mod2, testset, type="class")
> confusionMatrix(pred2, testset$classe)
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1673    6    0    0    0
         B    0 1131    4    0    0
         C    0    2 1020    4    0
         D    0    0    2  960    2
         E    1    0    0    0 1080

Overall Statistics

               Accuracy : 0.9964
                 95% CI : (0.9946, 0.9978)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9955
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9994   0.9930   0.9942   0.9959   0.9982
Specificity            0.9986   0.9992   0.9988   0.9992   0.9998
Pos Pred Value         0.9964   0.9965   0.9942   0.9959   0.9991
Neg Pred Value         0.9998   0.9983   0.9988   0.9992   0.9996
Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
Detection Rate         0.2843   0.1922   0.1733   0.1631   0.1835
Detection Prevalence   0.2853   0.1929   0.1743   0.1638
0.1837
Balanced Accuracy      0.9990   0.9961   0.9965   0.9975   0.9990
```

>> RF is better model then Decision tree. RF's Acurrancy overview 0.9964

So, I use RF model for Submission

What you should submit

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.

 # test model

predict(mod2, newdata = testing)

```
> predict(mod2, newdata = testing)
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
 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
```