Practical Machine Learning Project

Jiang Jin

October 24, 2015

Introduction

Modern portable devices like Jawbone Up, Nike FuelBand and Fitbit help people collecting a large amount of data about activity. A group of enthusiasts who take measurements about themselves regularly to improve their health, many of them trends to focus on how much of a particular activity they do, instead of how well they do these activities.

Data here collected accelerometers on the belt, forarm, arm and dumbell when the testers were asked to perform barbell lifts correctly and incorrectly in 5 different ways.
The project here is to use data perform prediction of how they did the exercise.

Data preprocessing & Modeling

Load Data:

```
setwd ("~/Desktop/WorkDirectory/Practical-Machine-Learning-Project")
library(caret)
```

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

```
library(rpart)
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(rattle)
```

```
## Loading required package: RGtk2
## Rattle: A free graphical interface for data mining with R.
## Version 3.5.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
# load data
training <- read.csv("./data/pml-training.csv", row.names=1)
dim (training)
```

```
## [1] 19622   159
```

```
testing <- read.csv("./data/pml-testing.csv", row.names=1)
dim (testing)
```

```
## [1]  20 159
```

Training and testing data set had 159 variables, training data had 19622 observations while testing data had 20 observations.

Data cleaning:

Here we remove the missing values and null values.

```
sum(complete.cases(training))
```

```
## [1] 406
```

```
training1 <- training[,colSums (is.na(training))==0]
testing1 <- testing [,colSums (is.na(testing))==0]
dim (training1); dim (testing1)
```

```
## [1] 19622    92
```

```
## [1] 20 59
```

```
# remove zero value
nsv <- nearZeroVar(training1, saveMetrics = TRUE)
train2 <- training1[,!nsv$nzv]
dim(train2)
```

```
## [1] 19622    58
```

```
nsv1 <- nearZeroVar (testing1, saveMetrics = TRUE)
test2 <- testing1[, !nsv1$nzv]
dim(test2)
```

```
## [1] 20 58
```

After data cleaning, training set contains 19622 observations and 58 variables, while testing set contains 20 observations and 58 variables.

Data Splitting:

Split training set into training and testing, 60% for training, 40% for cross validation.

```
inTrain <- createDataPartition(y=train2$classe, p=0.6, list=FALSE)
train <- train2 [inTrain,]
crossValidation <- train2[-inTrain,]

dim(train)
```

```
## [1] 11776    58
```

```
dim(crossValidation)
```

```
## [1] 7846   58
```

Modeling:

Use training part do modeling. “classe” is the factor to define activity type.

```
library(caret)
modFit <- train(classe ~., method="rf", data=train, trControl=trainControl(method='cv'), nu
mber=5, allowParallel=TRUE )
print (modFit$finalModel)
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, number = 5, allowParallel = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 40
##
##         OOB estimate of  error rate: 0.14%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3348    0    0    0    0 0.0000000000
## B    3 2273    3    0    0 0.0026327337
## C    0    2 2049    3    0 0.0024342746
## D    0    0    2 1926    2 0.0020725389
## E    0    0    0    2 2163 0.0009237875
```

Accuracy:

Compute accuracy in training set and cross validation set.

```
# train set
trainPre <- predict(modFit, train)
Matrix1 <- confusionMatrix (trainPre, train$classe)
Matrix1
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3348    0    0    0    0
##          B    0 2279    0    0    0
##          C    0    0 2054    0    0
##          D    0    0    0 1930    0
##          E    0    0    0    0 2165
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
# cross validation set
cvPre <- predict(modFit, crossValidation)
Matrix2 <- confusionMatrix (cvPre, crossValidation$classe)
Matrix2
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    3    0    0    0
##          B    0 1515    1    0    0
##          C    0    0 1365    0    0
##          D    0    0    2 1285    2
##          E    0    0    0    1 1440
##
## Overall Statistics
##
##                Accuracy : 0.9989
##                  95% CI : (0.9978, 0.9995)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9985
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9980   0.9978   0.9992   0.9986
## Specificity            0.9995   0.9998   1.0000   0.9994   0.9998
## Pos Pred Value         0.9987   0.9993   1.0000   0.9969   0.9993
## Neg Pred Value         1.0000   0.9995   0.9995   0.9998   0.9997
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1931   0.1740   0.1638   0.1835
## Detection Prevalence   0.2849   0.1932   0.1740   0.1643   0.1837
## Balanced Accuracy      0.9997   0.9989   0.9989   0.9993   0.9992
```

From the tet we can see the accuracy in training set is 1, P-value <0.05. Accuracy in cross validation set is 0.997, P-value <0.05. The accuracy is very high in method “rf”.

Test’s prediction

Here we use modFit to predict testing data set.

```
testPre <- predict (modFit, test2)
testPre
```

```
##  [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
```

```
summary(testPre)
```

```
## A B C D E
## 7 8 1 1 3
```

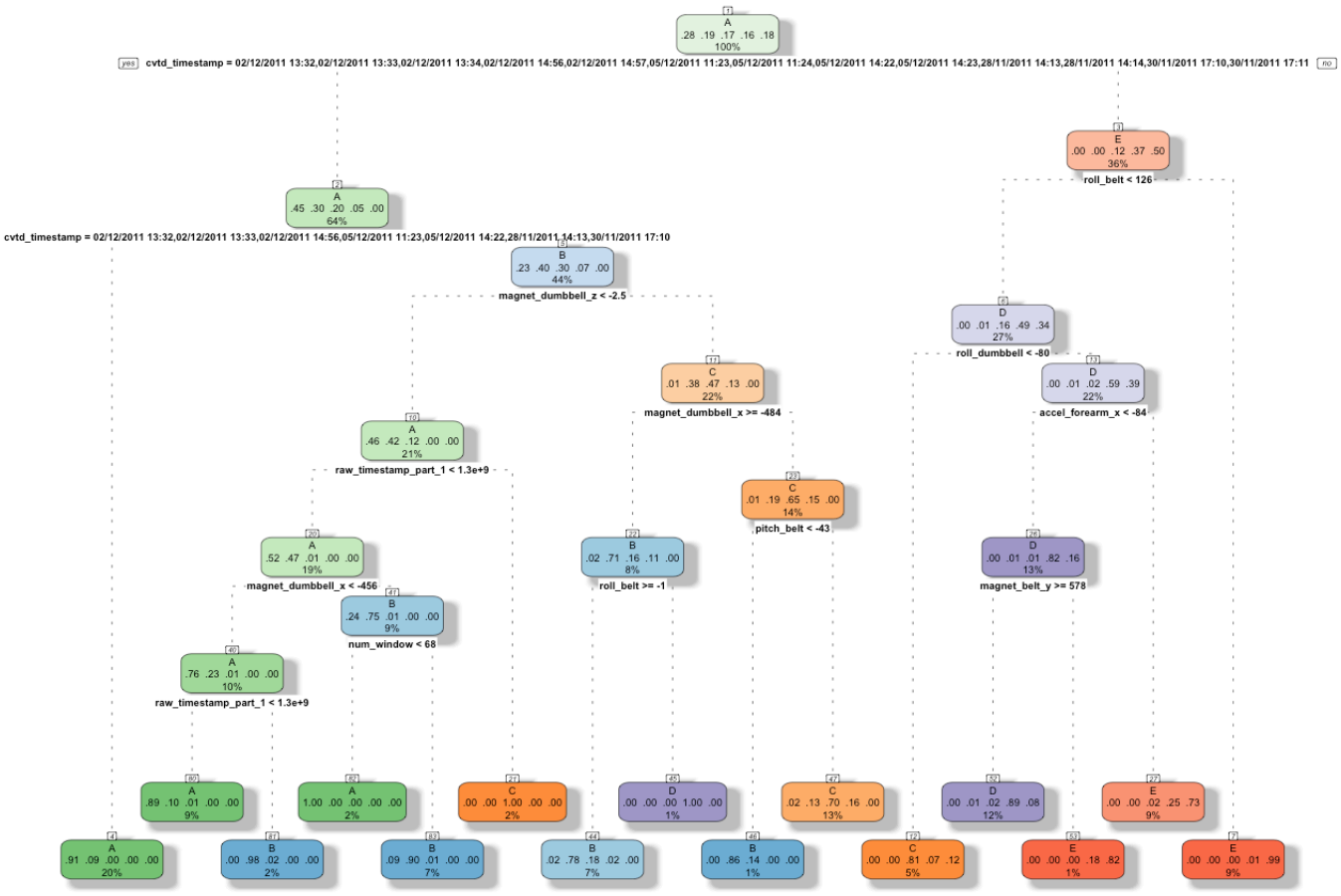
Result:

According to result, the 20 cases, 7 are class A, 8 are class B, 1 is class C, 1 is class D, 3 are class E.

Appendix:

plot desition tree

```
modFit2 <- rpart(classe ~ ., data=train, method="class")
library(rattle)
fancyRpartPlot(modFit2)
```



Rattle 2015-Oct-25 23:56:50 jinjiang