Course Project - Practical Machine Learning

Questions

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).

Data

The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Goal

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.

Model Building

1.Basic setting

Load the data.

```
data_training <- read.csv('pml-training.csv',sep = ",",header = T,na.strings=c("NA","#DIV/0!",""))
data_testing <- read.csv('pml-testing.csv',sep = ",",header = T,na.strings=c("NA","#DIV/0!",""))
data_training <- data_training[c(-1)]
data_testing <- data_testing[c(-1)]
```

Then delete variables with too much "NA" within (> 50%).

```
judge_factor <- rep(0,ncol(data_training)-1)
for(i in 1:(ncol(data_training)-1)){
        if(sum(is.na(data_training[,i]))/nrow(data_training) >= 0.5){
                judge_factor[i] <- 1
        }
}
id <- which(judge_factor == 0)
id <- c(id, ncol(data_training))
data_training <- data_training[, id]
data_testing <- data_testing[, id]
```

Split the dataset into the training set and the cv set.

```
library(caret)
```

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

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

```
inTrain <- createDataPartition(data_training$classe, p=0.6, list=FALSE)
data_train <- data_training[inTrain,]
data_cv <- data_training[-inTrain,]
```

2.Training

Choose the following method to train the algorithm:
- CART(method = "rpart");
- Linear Discriminant Analysis(method = "lda");
- Stochastic Gradient Boosting(method = "gbm").

```
set.seed(12345)
times0 <- Sys.time()
modelFit1 <- train(classe~., data=data_train, method="rpart")
times1 <- Sys.time()
modelFit2 <- train(classe~., data=data_train, method="lda")
times2 <- Sys.time()
modelFit3 <- train(classe~., data=data_train, method="gbm")
times3 <- Sys.time()
```

3.Evaluating the prediction accuracy and the time used for the training

```
prediction1 <- predict(modelFit1, newdata = data_cv, type = "raw")
prediction2 <- predict(modelFit2, newdata = data_cv, type = "raw")
prediction3 <- predict(modelFit3, newdata = data_cv, type = "raw")
confusionMatrix(prediction1, data_cv$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1654  362   37   53   17
##          B  248  889   15    0    0
##          C   17   39  737   39   80
##          D  304  228  579 1194  679
##          E    9    0    0    0  666
##
## Overall Statistics
##
##                Accuracy : 0.6551
##                  95% CI : (0.6445, 0.6656)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5668
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7410   0.5856  0.53874   0.9285  0.46186
## Specificity            0.9165   0.9584  0.97299   0.7271  0.99859
## Pos Pred Value         0.7791   0.7717  0.80811   0.4001  0.98667
## Neg Pred Value         0.8990   0.9060  0.90900   0.9811  0.89179
## Prevalence             0.2845   0.1935  0.17436   0.1639  0.18379
## Detection Rate         0.2108   0.1133  0.09393   0.1522  0.08488
## Detection Prevalence   0.2706   0.1468  0.11624   0.3803  0.08603
## Balanced Accuracy      0.8287   0.7720  0.75586   0.8278  0.73023
```

```
confusionMatrix(prediction2, data_cv$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2024  206    1    0    0
##          B  175 1125  148    0    0
##          C   32  182 1190  134    2
##          D    0    5   28 1094  140
##          E    1    0    1   58 1300
##
## Overall Statistics
##
##                Accuracy : 0.8581
##                  95% CI : (0.8502, 0.8658)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8206
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9068   0.7411   0.8699   0.8507   0.9015
## Specificity            0.9631   0.9490   0.9460   0.9736   0.9906
## Pos Pred Value         0.9072   0.7769   0.7727   0.8635   0.9559
## Neg Pred Value         0.9630   0.9386   0.9718   0.9708   0.9781
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2580   0.1434   0.1517   0.1394   0.1657
## Detection Prevalence   0.2843   0.1846   0.1963   0.1615   0.1733
## Balanced Accuracy      0.9350   0.8450   0.9079   0.9122   0.9461
```

```r
confusionMatrix(prediction3, data_cv$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    6    0    0    0
##          B    0 1509    2    0    0
##          C    0    3 1359   11    0
##          D    0    0    7 1275    6
##          E    0    0    0    0 1436
##
## Overall Statistics
##
##                Accuracy : 0.9955
##                  95% CI : (0.9938, 0.9969)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9944
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9941   0.9934   0.9914   0.9958
## Specificity            0.9989   0.9997   0.9978   0.9980   1.0000
## Pos Pred Value         0.9973   0.9987   0.9898   0.9899   1.0000
## Neg Pred Value         1.0000   0.9986   0.9986   0.9983   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1923   0.1732   0.1625   0.1830
## Detection Prevalence   0.2852   0.1926   0.1750   0.1642   0.1830
## Balanced Accuracy      0.9995   0.9969   0.9956   0.9947   0.9979
```

```r
ep1 <- difftime(times1, times0, units = "hour")
ep2 <- difftime(times2, times1, units = "hour")
ep3 <- difftime(times3, times2, units = "hour")
c(ep1, ep2, ep3)
```

```
## Time differences in hours
## [1] 0.02391543 0.01479224 1.99633058
```

Comparing the accuracy and time used, Linear Discriminant Analysis would be the best choice here. But in the predicting process after, three methods will all be used as reference.

4. Predict

```r
prediction_test1 <- predict(modelFit1, newdata = data_testing, type = "raw")
prediction_test2 <- predict(modelFit2, newdata = data_testing, type = "raw")
prediction_test3 <- predict(modelFit3, newdata = data_testing, type = "raw")
prediction_test <- cbind(prediction_test1, prediction_test2, prediction_test3)
prediction_test
```

```
##       prediction_test1 prediction_test2 prediction_test3
##  [1,]                4                 2                 2
##  [2,]                2                 2                 1
##  [3,]                2                 2                 2
##  [4,]                1                 1                 1
##  [5,]                1                 1                 1
##  [6,]                4                 5                 5
##  [7,]                4                 4                 4
##  [8,]                2                 3                 2
##  [9,]                1                 1                 1
## [10,]                1                 1                 1
## [11,]                2                 2                 2
## [12,]                3                 3                 3
## [13,]                2                 2                 2
## [14,]                1                 1                 1
## [15,]                4                 5                 5
## [16,]                4                 5                 5
## [17,]                4                 1                 1
## [18,]                2                 2                 2
## [19,]                4                 2                 2
## [20,]                2                 2                 2
```

According to the quiz results, the 2nd and the 8th predictions are wrong used the LDA method. So, the results are acceptable. Also, we can note that Stochastic Gradient Boosting method gives all right predictions, indicating that balancing the accuracy and the time used for training is hard.