Practical Machine Learning: Course Project

T.Y. 03-13-16

Introduction

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

Data

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har).

We download the two datasets

```
downloadcsv <- function(url, nastrings) {
    temp <- tempfile()
    download.file(url, temp, method = "curl")
    data <- read.csv(temp, na.strings = nastrings)
    unlink(temp)
    return(data)
}

trainurl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
train <- downloadcsv(trainurl, c("", "NA", "#DIV/0!"))

testurl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
test <- downloadcsv(testurl, c("", "NA", "#DIV/0!"))
```

The training data has 19622 observations and 160 features, and the distribution of the five measured stances A,B,C,D,E is:

```
dim(train)
```

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

```
table(train$classe)
```

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

Preprocessing

Partitioning the training set

We separate our training data into a training set and a validation set so that we can validate our model.

```
library(caret)
```

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

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

```
set.seed(123456)
trainset <- createDataPartition(train$classe, p = 0.8, list = FALSE)
Training <- train[trainset, ]
Validation <- train[-trainset, ]
```

Feature selection

First we clean up near zero variance features, columns with missing values and descriptive fields.

```
# exclude near zero variance features
nzvcol <- nearZeroVar(Training)
Training <- Training[, -nzvcol]

# exclude columns with m40% ore more missing values exclude descriptive
# columns like name etc
cntlength <- sapply(Training, function(x) {
        sum(!(is.na(x) | x == ""))
})
nullcol <- names(cntlength[cntlength < 0.6 * length(Training$classe)])
descriptcol <- c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_part_2",
                 "cvtd_timestamp", "new_window", "num_window")
excludecols <- c(descriptcol, nullcol)
Training <- Training[, !names(Training) %in% excludecols]
```

Model Train

We will use random forest as our model as implemented in the randomForest package by Breiman's random forest algorithm (based on Breiman and Cutler's original Fortran code) for classification and regression.

```
library(randomForest)
```

```
## 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
```

```
rfModel <- randomForest(classe ~ ., data = Training, importance = TRUE, ntrees = 10)
```

Model Validation

Let us now test our model performance on the training set itself and the cross validation set.

Training set accuracy

```
ptraining <- predict(rfModel, Training)
print(confusionMatrix(ptraining, Training$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4464    0    0    0    0
##          B    0 3038    0    0    0
##          C    0    0 2738    0    0
##          D    0    0    0 2573    0
##          E    0    0    0    0 2886
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9998, 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
```

Obviously our model performs excellent against the training set, but we need to cross validate the performance against the held out set and see if we have avoided overfitting.

Validation set accuracy (Out-of-Sample)

Let us now see how our model performs on the cross validation set that we held out from training.

```
library(e1071)
pvalidation <- predict(rfModel, Validation)
print(confusionMatrix(pvalidation, Validation$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    7    0    0    0
##          B    0  751    4    0    0
##          C    0    1  680    4    0
##          D    0    0    0  639    4
##          E    0    0    0    0  717
##
## Overall Statistics
##
##                Accuracy : 0.9949
##                  95% CI : (0.9921, 0.9969)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9936
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9895   0.9942   0.9938   0.9945
## Specificity            0.9975   0.9987   0.9985   0.9988   1.0000
## Pos Pred Value         0.9938   0.9947   0.9927   0.9938   1.0000
## Neg Pred Value         1.0000   0.9975   0.9988   0.9988   0.9988
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1914   0.1733   0.1629   0.1828
## Detection Prevalence   0.2863   0.1925   0.1746   0.1639   0.1828
## Balanced Accuracy      0.9988   0.9941   0.9963   0.9963   0.9972
```

The cross validation accuracy is 99.5% and the out-of-sample error is therefore 0.5% so our model performs rather good.

Test set prediction

The prediction of our algorithm for the test set is:

````
    ```r
    ptest <- predict(rfModel, test)
    ptest
    ```

    ```
    ##  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
    ```
````

Conclusion

The model predicted the 20 test cases with 100% accuracy. All 20 points were awarded after submitting the 20 test files.