Coursera_machineLearning

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

analyzing

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(caret)
```

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

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

```
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
```

```
set.seed(2017)

# loading and training and testing data
trainingdata    <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!", ""
))
testingdata     <- read.csv('pml-testing.csv', na.strings=c("NA","#DIV/0!", "")
)

# Check dimensions for number of variables and number of observations
dim(trainingdata)
```

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

```
dim(testingdata)
```

```
## [1]  20 160
```

feature selection

There are 160 columns of data in both testing and training datasets. Let's check if the columns are the same:

```
setdiff(colnames(testingdata),colnames(trainingdata))
```

```
## [1] "problem_id"
```

```
setdiff(colnames(trainingdata),colnames(testingdata))
```

```
## [1] "classe"
```

The columns are the same, except for problem_id in testing data and classe in training data. Some data is irrelevant to this analysis: for example, the “user_name”, and time of testing are irrelevant to the model. This includes the first 7 columns.

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

Also removing the features that are all NAs:

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

including_features <- intersect(colnames(trainingdata),colnames(trainingdata))
```

The includeing features are: `print(including_features)`

building the models with cross-validation

Training datset is segmented into training and testing for cross-validation.

```
folds       <- createDataPartition(y=trainingdata$classe, p=0.75, list=FALSE)
training    <- trainingdata[folds, ]
validating  <- trainingdata[-folds, ]
table(training$classe)
```

```
##
##    A    B    C    D    E
## 4185 2848 2567 2412 2706
```

modeling

Model the classe using random forest:

```
rfModel <- randomForest(classe ~ ., data = training, importance = TRUE)
predicted <- predict(rfModel, training)
print(confusionMatrix(predicted, training$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4185    0    0    0    0
##          B    0 2848    0    0    0
##          C    0    0 2567    0    0
##          D    0    0    0 2412    0
##          E    0    0    0    0 2706
##
## 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   0.1839
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1839
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1839
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
predicted <- predict(rfModel, validating)
print(confusionMatrix(predicted, validating$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    7    0    0    0
##          B    0  941    3    0    0
##          C    0    1  851   11    1
##          D    0    0    1  793    3
##          E    1    0    0    0  897
##
## Overall Statistics
##
##                Accuracy : 0.9943
##                  95% CI : (0.9918, 0.9962)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9928
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9916   0.9953   0.9863   0.9956
## Specificity            0.9980   0.9992   0.9968   0.9990   0.9998
## Pos Pred Value         0.9950   0.9968   0.9850   0.9950   0.9989
## Neg Pred Value         0.9997   0.9980   0.9990   0.9973   0.9990
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1919   0.1735   0.1617   0.1829
## Detection Prevalence   0.2857   0.1925   0.1762   0.1625   0.1831
## Balanced Accuracy      0.9986   0.9954   0.9961   0.9927   0.9977
```

The classifier has high validation performance: 99%.

predicting classe of testing data set.

```
predicted_classe <- predict(rfModel, testingdata)
predicted_classe
```

```
##  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
```