Practical Machine Learning Course Project

Xiangting

Sunday, August 23, 2015

Project 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 Sets

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.

Project Objective

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.

Analysis

Approach

1. Reproducibility: Launch packages and set seed
2. Load and explore the data set
3. Cross-validation: 70% of the original data is used for model building (training data) while the rest of 30% of the data is used for testing (testing data)
4. Data cleaning: Remove variables which have little predicitive value or information.
5. Apply PCA to reduce the number f variables
6. Apply random forest method to build a model
7. Check the model with the testing data set
8. Apply the model to estimate classes of 20 observations

Reproducibility: Launch packages and set seed

```
set.seed(1111)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.1
```

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

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.2.2
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.2.2
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

Load and explore the data set

```
#Results hidden
data <- read.csv("C:/Users/User1/Documents/GitHub/pml-training.csv", na.strings=c("NA","#DI
V/0!", ""))
colnames(data)
summary(data)
```

Cross-validation : 70% training data, 30% testing data

```
train <- createDataPartition(y=data$classe,p=.70,list=F)
training <- data[train,]
testing <- data[-train,]
```

Data Cleaning

```
#Results hidden
#Remove variables which have little predicitive value: identifier, timestamp, and window data
Cl <- grep("name|timestamp|window|X", colnames(training), value=F)
trainingCl <- training[,-Cl]

#Remove variables with over 95% missing data
trainingCl[trainingCl==""] <- NA
NArate <- apply(trainingCl, 2, function(x) sum(is.na(x)))/nrow(trainingCl)
trainingCl <- trainingCl[!(NArate>0.95)]
summary(trainingCl)
```

Apply PCA to reduce the number f variables

```
preProc <- preProcess(trainingCl[,1:52],method="pca",thresh=.8) #12 components are required
preProc
```

```
##
## Call:
## preProcess.default(x = trainingCl[, 1:52], method = "pca", thresh = 0.8)
##
## Created from 13737 samples and 52 variables
## Pre-processing: principal component signal extraction, scaled, centered
##
## PCA needed 12 components to capture 80 percent of the variance
```

```
preProc <- preProcess(trainingCl[,1:52],method="pca",thresh=.9) #18 components are required
preProc
```

```
##
## Call:
## preProcess.default(x = trainingCl[, 1:52], method = "pca", thresh = 0.9)
##
## Created from 13737 samples and 52 variables
## Pre-processing: principal component signal extraction, scaled, centered
##
## PCA needed 18 components to capture 90 percent of the variance
```

```
preProc <- preProcess(trainingCl[,1:52],method="pca",thresh=.95) #25 components are required
preProc
```

```
##
## Call:
## preProcess.default(x = trainingCl[, 1:52], method = "pca", thresh = 0.95)
##
## Created from 13737 samples and 52 variables
## Pre-processing: principal component signal extraction, scaled, centered
##
## PCA needed 25 components to capture 95 percent of the variance
```

```
preProc <- preProcess(trainingCl[,1:52],method="pca",pcaComp=25) #Use 25 components to achieve 9
5% of variance
preProc
```

```
##
## Call:
## preProcess.default(x = trainingCl[, 1:52], method = "pca", pcaComp = 25)
##
## Created from 13737 samples and 52 variables
## Pre-processing: principal component signal extraction, scaled, centered
##
## PCA used 25 components as specified.
```

```
preProc$rotation
trainingPC <- predict(preProc,trainingCl[,1:52])
```

Apply random forest method to build a model

```
modFitRF <- randomForest(trainingCl$classe ~ .,   data=trainingPC, do.trace=F)
print(modFitRF) # view results
```

```
##
## Call:
##  randomForest(formula = trainingCl$classe ~ ., data = trainingPC,      do.trace = F)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 2.53%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3865   14   12   10    5  0.01049667
## B   39 2576   37    1    5  0.03085026
## C    4   39 2325   23    5  0.02963272
## D    5    4   99 2140    4  0.04973357
## E    1    9   16   15 2484  0.01623762
```

```
importance(modFitRF) # importance of each predictor
```

```
##      MeanDecreaseGini
## PC1          594.5547
## PC2          471.9692
## PC3          490.8214
## PC4          367.5061
## PC5          554.0366
## PC6          431.5030
## PC7          419.3840
## PC8          686.6000
## PC9          493.5188
## PC10         395.3707
## PC11         350.1926
## PC12         598.7993
## PC13         355.7581
## PC14         643.4628
## PC15         477.3593
## PC16         446.9813
## PC17         422.6061
## PC18         288.0795
## PC19         346.2517
## PC20         359.9390
## PC21         401.4056
## PC22         411.1930
## PC23         242.3747
## PC24         253.1280
## PC25         357.0318
```

Check the model with the testing data set

```
testingCl <- testing[,-Cl]
testingCl[testingCl==""] <- NA
NArate <- apply(testingCl, 2, function(x) sum(is.na(x)))/nrow(testingCl)
testingCl <- testingCl[!(NArate>0.95)]
testingPC <- predict(preProc,testingCl[,1:52])
confusionMatrix(testingCl$classe,predict(modFitRF,testingPC))
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1661    2    7    2    2
##          B   12 1109   16    0    2
##          C    0   13 1003   10    0
##          D    1    2   40  921    0
##          E    0    3   12    4 1063
## 
## Overall Statistics
## 
##                Accuracy : 0.9782
##                  95% CI : (0.9742, 0.9818)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9725
##  Mcnemar's Test P-Value : 7.489e-08
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9922   0.9823   0.9304   0.9829   0.9963
## Specificity            0.9969   0.9937   0.9952   0.9913   0.9961
## Pos Pred Value         0.9922   0.9737   0.9776   0.9554   0.9824
## Neg Pred Value         0.9969   0.9958   0.9846   0.9967   0.9992
## Prevalence             0.2845   0.1918   0.1832   0.1592   0.1813
## Detection Rate         0.2822   0.1884   0.1704   0.1565   0.1806
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9946   0.9880   0.9628   0.9871   0.9962
```

Apply the model to estimate classes of 20 observations

```
testdata <- read.csv("C:/Users/User1/Documents/GitHub/pml-testing.csv", na.strings=c("NA","#DI
V/0!", ""))
testdataCl <- testdata[,-Cl]
testdataCl[testdataCl==""] <- NA
NArate <- apply(testdataCl, 2, function(x) sum(is.na(x)))/nrow(testdataCl)
testdataCl <- testdataCl[!(NArate>0.95)]
testdataPC <- predict(preProc,testdataCl[,1:52])
testdataCl$classe <- predict(modFitRF,testdataPC)
```

Write files for submission

```
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}

pml_write_files(testdataCl$classe)
```

Conclusion

To analyze and predict correct body movement during the exercise, 19622 observations from weight lifting exercise were collected. These observations are randomly partitioned to 2 sets: 70% (13737 observations) to build a model by random forest method, and the remaining 30% (5885 observations) to be used as the testing set for model validation (cross-validation).

The model statistics showed that the model had the overall accuracy of 97% for the testing set. The sensitivity is 92%-99% and the specificity was over 99% for all classes.

Limitation of study: The observation data used in the analyses was collected from 6 young health participants in an experiment using Microsoft Kinect. Under the same condition, the model is expected to perform over 95% accuracy. However, under different conditions (e.g. elderly people or different measuring device) e, the model might not perform as well.