course\_project\_ML

Ananya Tripathi

June 30, 2016

# Introduction

The goal of the project is to predict the manner in which the participants did the exercise which is shown by the classe variable.The data for this project comes from:<http://groupware.les.inf.puc-rio.br/har.>

## Required Packages

library(data.table)  
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

library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:data.table':  
##   
## hour, mday, month, quarter, wday, week, yday, year

## The following object is masked from 'package:base':  
##   
## date

setwd("C:/PNY")  
r\_data <- fread("pml-training.csv")  
rt <- fread("pml-testing.csv")  
r\_data <- as.data.frame(r\_data)  
inTrain <- createDataPartition(r\_data$classe, p=0.8, list = FALSE)  
r <- r\_data[inTrain,]  
r\_test <- r\_data[-inTrain,]

## Removing columns

We now find the columns with NA,#DIV/0! and "" values and remove from the training data set

sum\_na <- apply(r,2,function(x){((sum((is.na(x))\*1)))})   
index\_na <- which(sum\_na!=0)  
names(index\_na) <- NULL  
  
sum\_txt <- apply(r,2,function(x){((sum((x=="#DIV/0!")\*1)))})   
index\_txt <- which(sum\_txt!=0)  
names(index\_txt)<- NULL  
  
sum\_empty <- apply(r,2,function(x){((sum((as.character(x)=="")\*1)))})   
index\_empty <- which(sum\_empty!=0)  
names(index\_empty)<- NULL  
  
input <- unique(c(index\_na,index\_txt,index\_empty))  
  
read <- r[,-input]

## More cleaning

Now the columns with redundant values are removed, the classe column is converted to a factor variable,day of the week and time is extracted from "cvtd\_timestamp" using lubridate package and a new data frame "data" is made. The classe column is removed before preprocessing the data.

read$classe <- as.factor(read$classe)  
read\_final <- read[,-c(1,2,6)]  
  
library(lubridate)  
  
date <- dmy(substr(r$cvtd\_timestamp,1,10))  
time <- hm(substr(r$cvtd\_timestamp,12,16))  
day <- wday(date)  
hour <- hour(time)  
data <- data.frame(read\_final[,-c(3,57)],day,hour)

## Model Building

In the next step, principal component analysis is carried out to make new columns which explain most variation in the dataset prepared till now. We then apply the random forest model and carry out cross validation with a value of 5. The model generated shows that the final value used for the model was mtry = 2, which generated the accuracy of 0.9723609.

prep <- preProcess(data,method="pca")  
data\_prep <- predict(prep,data)  
classe <- read\_final$classe  
mdl\_data <- data.frame(classe,data\_prep)  
  
control <- trainControl(method = 'cv', number = 5)  
model\_rf <- train(classe~.,method="rf",trControl = control, data=mdl\_data)  
model\_rf

## Random Forest   
##   
## 15699 samples  
## 26 predictor  
## 5 classes: 'A', 'B', 'C', 'D', 'E'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold)   
## Summary of sample sizes: 12558, 12557, 12560, 12561, 12560   
## Resampling results across tuning parameters:  
##   
## mtry Accuracy Kappa   
## 2 0.9784707 0.9727646  
## 14 0.9737569 0.9667992  
## 26 0.9645206 0.9551092  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 2.

The in sample error rate can be seen from the confusion matrix generated.There is no classification error for the training model.

confusionMatrix(predict(model\_rf),r$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

## Predicting on r\_test data to find out of sample error

Then test data taken from the pml\_training is read. As the columns to be removed are already calculated for train data, same columns can be removed from test data. Weekday and time is extracted from the test data, a new data frame is made, preprocessing is done using the pca model of the TRAIN data and prediction is carried out using the model for train data.

readt <- r\_test[,-input]  
readt\_final <- readt[,-c(1,2,6)]  
  
tdate <- dmy(substr(r\_test$cvtd\_timestamp,1,10))  
ttime <- hm(substr(r\_test$cvtd\_timestamp,12,16))  
day <- wday(tdate)  
hour <- hour(ttime)  
tdata <- data.frame(readt\_final[,-c(3,57)],day,hour)  
tdata\_prep <- predict(prep,tdata)  
  
model\_test\_rf <- predict(model\_rf,tdata\_prep)  
confusionMatrix(model\_test\_rf, r\_test$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1114 10 0 2 0  
## B 1 738 8 1 2  
## C 0 11 670 23 0  
## D 1 0 6 616 1  
## E 0 0 0 1 718  
##   
## Overall Statistics  
##   
## Accuracy : 0.9829   
## 95% CI : (0.9784, 0.9867)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9784   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9982 0.9723 0.9795 0.9580 0.9958  
## Specificity 0.9957 0.9962 0.9895 0.9976 0.9997  
## Pos Pred Value 0.9893 0.9840 0.9517 0.9872 0.9986  
## Neg Pred Value 0.9993 0.9934 0.9957 0.9918 0.9991  
## Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838  
## Detection Rate 0.2840 0.1881 0.1708 0.1570 0.1830  
## Detection Prevalence 0.2870 0.1912 0.1795 0.1591 0.1833  
## Balanced Accuracy 0.9970 0.9843 0.9845 0.9778 0.9978

The out of sample accuracy comes out to be 0.985. Hence, the out of sample error is 1.5%

## Prediction on test data

rt <- as.data.frame(rt)  
rt <- rt[,-160]  
readT <- rt[,-input]  
readT\_final <- readT[,-c(1,2,6)]  
  
Tdate <- dmy(substr(rt$cvtd\_timestamp,1,10))  
Ttime <- hm(substr(rt$cvtd\_timestamp,12,16))  
day <- wday(Tdate)  
hour <- hour(Ttime)  
Tdata <- data.frame(readT\_final[,-c(3)],day,hour)  
Tdata\_prep <- predict(prep,Tdata)

## Submission of the predicted answers using Coursera provided code.

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)  
 }  
}  
  
model\_Test\_rf <- predict(model\_rf,Tdata\_prep)  
  
model\_Test\_rf

## [1] B A A A A E D B A A B C B A E E A B B B  
## Levels: A B C D E

pml\_write\_files(model\_Test\_rf)