Pactical Machine Learning Project Report

Ravi Kumar Yadav

## Executive Summary

This document is the final report of the Peer Assessment project from Coursera course Practical Machine Learning, as part of the Specialization in Data Science. Given both training and test data from the following study:

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

The goal of this project is to “predict the manner in which they did the exercise.” This report would describes:

“how model is build” “how cross validation is performed for model evaluation” “what is the expected out of sample error” “Reasons for the choices made”

Ultimately “classe” variable needs to be successfully predicted using machine learning algorithms. The machine learning algorithm described here is applied to the 20 test cases available in the test data and the predictions are submitted in appropriate format to the Course Project Prediction Quiz for automated grading.

First load the required packages and set the seed for reproducibility:

library(lubridate)

## Warning: package 'lubridate' was built under R version 3.2.5

##   
## Attaching package: 'lubridate'

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

library(caret)

## Warning: package 'caret' was built under R version 3.2.5

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.2.5

library(ade4)

## Warning: package 'ade4' was built under R version 3.2.5

library(utiml)

## Loading required package: mldr

##   
## Attaching package: 'utiml'

## The following object is masked from 'package:caret':  
##   
## lift

library(nnet)  
library(e1071)

## Warning: package 'e1071' was built under R version 3.2.5

set.seed(12345)

## Data Loading and Cleaning

Data is loaded from local machine. Next data cleaning is perfromed; all the blank and '#DIV/0' were converted to NAs for both test and train data.

train <- read.csv("pml-training.csv",stringsAsFactors = F)  
test <- read.csv("pml-testing.csv",stringsAsFactors = F)  
  
for(x in 1:length(train)){  
 train[train[,x] %in% c('','#DIV/0'),x] <- NA  
}  
  
for(x in 1:length(test)){  
 test[test[,x] %in% c('','#DIV/0'),x] <- NA  
}

Next step is dealing with NAs, Looking into the data I foud out that there are colums with either complete NAs or no NAs. So chose to remove NAs colume wise using a simple function:

remove\_cols <- function(x,n = 0.5){  
 miss\_col <- c()  
 for(i in 1:ncol(x)) {  
 if(length(which(is.na(x[,i]))) > n\*nrow(x)) miss\_col <- append(miss\_col,i)   
 }  
 x <- x[,- miss\_col]  
   
 return(x)  
}  
train <- remove\_cols(train,n = 0.3)  
test <- remove\_cols(test,n = 0.3)

Now little bit knowhow about the data made me remove the timestamp column. And ofcourse index column needs to be removed.

train$cvtd\_timestamp <- test$cvtd\_timestamp <- NULL  
train$X <- test$X <- NULL

Created factor and numeric colums:

train\_char\_index <- sapply(train,is.character)  
test\_char\_index <- sapply(test,is.character)  
  
train[train\_char\_index] <- lapply(train[train\_char\_index],factor)  
test[train\_char\_index] <- lapply(test[train\_char\_index],factor)

Now we have the data ready in right fromat. Now lets remove any highly corelated variable for better accuracy. Numeric variables were also tested for zero variance, however no such variable were found.

train\_num\_index <- sapply(train,is.numeric)  
train\_fac\_index <- sapply(train,is.factor)  
train\_num <- train[train\_num\_index]  
train\_fac <- train[train\_fac\_index]  
df <- cor(train\_num)  
hc <- findCorrelation(df, cutoff=0.9) # putt any value as a "cutoff"   
hc <- sort(hc)  
train\_num <- train\_num[,-c(hc)]

After cleaning the train data, ensured that test data also contains the same variable as train data:

train <- as.data.frame(cbind(data.frame(train\_num),data.frame(train\_fac)))  
test <- test[,colnames(test) %in% colnames(train)]  
test <- test[,c(colnames(train[,-length(train)]))]

## Model Building

First of all splitted the train data into testing and training data in 80:20 ratiO. Based on my previous experince with multiclass classification problem, I chose to use use Binary relevance algorithm. For BR algorithm data needs to be converted into mldr format.

Model performance is evaluated with confusion matrix; showing above 99% accuracy. The result is very good however to gain more confidance into the model cross validation is performed as a next step.

intrain <- createDataPartition(train$classe,p = 0.8,list = FALSE)  
training <- train[intrain,]  
testing <- train[-intrain,]  
  
indx\_f <- sapply(training, is.factor)  
indx\_n <- sapply(training, is.numeric)  
indx\_c <- sapply(training, is.character)  
testd <- acm.disjonctif(training[indx\_f]) # converts factor variables into dummy binary variables  
training <- cbind(training[indx\_c],training[indx\_n],testd)  
  
indx\_f <- sapply(testing, is.factor)  
indx\_n <- sapply(testing, is.numeric)  
indx\_c <- sapply(testing, is.character)  
testd <- acm.disjonctif(testing[indx\_f]) # converts factor variables into dummy binary variables  
testing <- cbind(testing[indx\_c],testing[indx\_n],testd)  
  
mymldr <- mldr\_from\_dataframe(training, labelIndices = c((length(training)-4):length(training)), name = "trainMLDR")  
mymldr\_test <- mldr\_from\_dataframe(testing, labelIndices = c((length(testing)-4):length(testing)), name = "testMLDR")  
  
model <- br(mymldr, "RF", seed = 123)  
  
pred <- as.data.frame(as.probability(predict(model, mymldr\_test)))  
  
actual <- testing[,(length(testing)-4):length(testing)]  
testing$actual <- factor(apply(actual, 1, function(x) which.is.max(x)), labels = colnames(actual))  
testing$prediction <- factor(apply(pred, 1, function(x) which.is.max(x)), labels = colnames(pred))  
  
xtab <- table(testing$actual,testing$prediction)  
confusionMatrix(xtab)

## Confusion Matrix and Statistics  
##   
##   
## classe.A classe.B classe.C classe.D classe.E  
## classe.A 1116 0 0 0 0  
## classe.B 0 759 0 0 0  
## classe.C 0 4 680 0 0  
## classe.D 0 0 3 640 0  
## classe.E 0 0 0 0 721  
##   
## Overall Statistics  
##   
## Accuracy : 0.9982   
## 95% CI : (0.9963, 0.9993)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9977   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: classe.A Class: classe.B Class: classe.C  
## Sensitivity 1.0000 0.9948 0.9956  
## Specificity 1.0000 1.0000 0.9988  
## Pos Pred Value 1.0000 1.0000 0.9942  
## Neg Pred Value 1.0000 0.9987 0.9991  
## Prevalence 0.2845 0.1945 0.1741  
## Detection Rate 0.2845 0.1935 0.1733  
## Detection Prevalence 0.2845 0.1935 0.1744  
## Balanced Accuracy 1.0000 0.9974 0.9972  
## Class: classe.D Class: classe.E  
## Sensitivity 1.0000 1.0000  
## Specificity 0.9991 1.0000  
## Pos Pred Value 0.9953 1.0000  
## Neg Pred Value 1.0000 1.0000  
## Prevalence 0.1631 0.1838  
## Detection Rate 0.1631 0.1838  
## Detection Prevalence 0.1639 0.1838  
## Balanced Accuracy 0.9995 1.0000

cnfm <- multilabel\_confusion\_matrix(mymldr\_test,predict(model, mymldr\_test))  
print(cnfm)

## Multi-label Confusion Matrix  
##   
## Absolute Matrix:  
## -------------------------------------  
## Expected\_1 Expected\_0 TOTAL  
## Prediction\_1 3916 7 3923  
## Predicion\_0 7 15685 15692  
## TOTAL 3923 15692 19615  
##   
## Proportinal Matrix:  
## -------------------------------------  
## Expected\_1 Expected\_0 TOTAL  
## Prediction\_1 0.2 0.0 0.2  
## Predicion\_0 0.0 0.8 0.8  
## TOTAL 0.2 0.8 1.0  
##   
## Label Matrix  
## -------------------------------------  
## TP FP FN TN Correct Wrong %TP %FP %FN %TN %Correct %Wrong  
## classe.A 1116 0 0 2807 3923 0 0.28 0 0 0.72 1 0  
## classe.B 759 4 0 3160 3919 4 0.19 0 0 0.81 1 0  
## classe.C 680 3 4 3236 3916 7 0.17 0 0 0.82 1 0  
## classe.D 640 0 3 3280 3920 3 0.16 0 0 0.84 1 0  
## classe.E 721 0 0 3202 3923 0 0.18 0 0 0.82 1 0  
## MeanRanking MeanScore  
## classe.A 2.90 0.28  
## classe.B 2.50 0.19  
## classe.C 3.04 0.17  
## classe.D 2.97 0.16  
## classe.E 3.59 0.18

## Model evaluation with Cross Validation

folds <- create\_kfold\_partition(mymldr, k = 10)  
 for (i in 1:10) {  
 dataset <- partition\_fold(folds, i)  
 training <- dataset$train  
 testing <- dataset$test  
 model\_split <- br(training, 'RF')  
 pred\_split <- predict(model\_split, testing)  
 cnfm\_split <- multilabel\_confusion\_matrix(testing,pred\_split)  
 if(i == 1){  
 cnfm\_kfold <- cnfm\_split  
 }  
 else{  
 cnfm\_kfold <- "+"(cnfm\_kfold,cnfm\_split)  
 }  
 }  
print(cnfm\_kfold)

## Multi-label Confusion Matrix  
##   
## Absolute Matrix:  
## -------------------------------------  
## Expected\_1 Expected\_0 TOTAL  
## Prediction\_1 15684 19 15703  
## Predicion\_0 15 62777 62792  
## TOTAL 15699 62796 78495  
##   
## Proportinal Matrix:  
## -------------------------------------  
## Expected\_1 Expected\_0 TOTAL  
## Prediction\_1 0.2 0.0 0.2  
## Predicion\_0 0.0 0.8 0.8  
## TOTAL 0.2 0.8 1.0  
##   
## Label Matrix  
## -------------------------------------  
## TP FP FN TN Correct Wrong %TP %FP %FN %TN %Correct %Wrong  
## classe.A 4464 2 0 11233 15697 2 0.28 0 0 0.72 1 0  
## classe.B 3036 3 2 12658 15694 5 0.19 0 0 0.81 1 0  
## classe.C 2733 7 5 12954 15687 12 0.17 0 0 0.83 1 0  
## classe.D 2566 4 7 13122 15688 11 0.16 0 0 0.84 1 0  
## classe.E 2885 3 1 12810 15695 4 0.18 0 0 0.82 1 0  
## MeanRanking MeanScore  
## classe.A 2.92 0.28  
## classe.B 2.50 0.19  
## classe.C 3.03 0.17  
## classe.D 2.95 0.16  
## classe.E 3.59 0.18

Cross validation result again showed more than 99% accuracy(summation of %true positives for all classes). So now we can belive more on the model.

## Prediction on given test sample

With the build model prediction is made and result is reported:

indx\_f <- sapply(test, is.factor)  
indx\_n <- sapply(test, is.numeric)  
indx\_c <- sapply(test, is.character)  
testd <- acm.disjonctif(test[indx\_f])  
test <- cbind(test[indx\_c],test[indx\_n],testd)  
test$new\_window.yes <- 0  
  
pred\_test <- as.data.frame(as.probability(predict(model, test)))  
  
test$prediction <- factor(apply(pred\_test, 1, function(x) which.is.max(x)), labels = colnames(pred\_test))  
  
test$prediction <- gsub("classe.","",test$prediction)  
test$prediction <- as.factor(test$prediction)  
  
test$prediction

## [1] 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