**Practical Machine Learning**

For this assignment I analyzed the provided data to determine what activity an individual perform. To do this I made use of caret and randomForest, this allowed me to generate correct answers for each of the 20 test data cases provided in this assignment. I made use of a seed value for consistent results.

library(Hmisc)

library(caret)

library(randomForest)

library(foreach)

library(doParallel)

set.seed(2048)

options(warn=-1)

First, I loaded the data both from the provided training and test data provided by Coursera Some values contained a "#DIV/0!" that I replaced with an NA value.

training\_data <- read.csv("pml-training.csv", na.strings=c("#DIV/0!") )

evaluation\_data <- read.csv("pml-testing.csv", na.strings=c("#DIV/0!") )

I also casted all columns 8 to the end to be numeric.

for(i in c(8:ncol(training\_data)-1)) {training\_data[,i] = as.numeric(as.character(training\_data[,i]))}

for(i in c(8:ncol(evaluation\_data)-1)) {evaluation\_data[,i] = as.numeric(as.character(evaluation\_data[,i]))}

Some columns were mostly blank. These did not contribute well to the prediction. I chose a feature set that only included complete columns. We also remove user name, timestamps and windows.

Determine and display out feature set.

feature\_set <- colnames(training\_data[colSums(is.na(training\_data)) == 0])[-(1:7)]

model\_data <- training\_data[feature\_set]

feature\_set

We now have the model data built from our feature set.

idx <- createDataPartition(y=model\_data$classe, p=0.75, list=FALSE )

training <- model\_data[idx,]

testing <- model\_data[-idx,]

We now build 5 random forests with 150 trees each. We make use of parallel processing to build this model. I found several examples of how to perform parallel processing with random forests in R, this provided a great speedup.

registerDoParallel()

x <- training[-ncol(training)]

y <- training$classe

rf <- foreach(ntree=rep(150, 6), .combine=randomForest::combine, .packages='randomForest') %dopar% {

randomForest(x, y, ntree=ntree)

}

Provide error reports for both training and test data.

predictions1 <- predict(rf, newdata=training)

confusionMatrix(predictions1,training$classe)

predictions2 <- predict(rf, newdata=testing)

confusionMatrix(predictions2,testing$classe)

## Conclusions and Test Data Submit

As can be seen from the confusion matrix this model is very accurate. I did experiment with PCA and other models, but did not get as good of accuracy. Because my test data was around 99% accurate I expected nearly all of the submitted test cases to be correct. It turned out they were all correct.

Prepare the submission. (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)

}

}

x <- evaluation\_data

x <- x[feature\_set[feature\_set!='classe']]

answers <- predict(rf, newdata=x)

answers

pml\_write\_files(answers)