Practical machine Learning Assignmentt4

knitr::opts\_chunk$set(echo = TRUE)

# practical machine learning assignment week4

# brief data description

This data is about various observations for weight lifters made possible through wearable gadgets The objective is to use the data to predict whether the weight lifting is done the right way or the wrong way based on the model developed

# Step 1: Read Training data to build the model

train=read.csv("wtlift\_train.csv",na.strings=c("NA","#DIV/0!", ""))

# step 2: Remove columns having na

train1=train[,colSums(is.na(train))==0]

# step 3: remove the first 7 columns which are irrelevant for the analysis

library(dplyr)  
train11=select(train1,-c(1:7))

# step 4: split the train data into intrain and intest for cross validation

library(caret)  
intrainidx=createDataPartition(train11$classe,p=0.75)[[1]]  
intrain=train11[intrainidx,]  
intest=train11[-intrainidx,]

# step 5: fit decision tree to the intrain data

fittree=train(classe~.,method="rpart",data=intrain)

# step 6: predict for intest based on the decision tree model

prdtree=predict(fittree,intest)

# step 7: compute accuracy for the decision tree on intest dataset

confusionMatrix(intest$classe,prdtree)$overall['Accuracy']

## Accuracy   
## 0.5

# step 8: overall accuracy for decision tree is very low. Hence now fitting gradient boosting machines

library(gbm)  
fitgbm=gbm(classe~.,data=intrain)

## Distribution not specified, assuming multinomial ...

prdgbm=predict(fitgbm,intest,type='response')  
prdgbm\_class=apply(prdgbm,1,which.max)  
a=as.factor(prdgbm\_class)  
levels(a)=c("A","B","C","D","E")  
confusionMatrix(intest$classe,a)$overall["Accuracy"]

## Accuracy   
## 0.8138254

# step 9: Accuracy has improved. Now fitting random forest to see if accuracy improves further.

library(randomForest)  
fitrf=randomForest(classe~.,method="class",data=intrain)  
prdrf=predict(fitrf,intest,type="class")  
confusionMatrix(intest$classe,prdrf)$overall["Accuracy"]

## Accuracy   
## 0.9961256

expected\_error=as.numeric(1-confusionMatrix(intest$classe,prdrf)$overall["Accuracy"])  
print(paste0("expected error " , signif(expected\_error,digits=5)))

## [1] "expected error 0.0038744"

# step 10: selecting random forest as the accuracy has now further improved .

# Using the random Forest model predictor developed on intrain dataset to predict the 20 outcomes in the Test dataset

test=read.csv("wtlift\_testing.csv",na.strings=c("NA","#DIV/0!", ""))  
test1=test[,colSums(is.na(train))==0]  
test11=select(test1,-c(1:7))  
prdrffinal=predict(fitrf,test11,type="class")  
prdrffinal

# Conclusion: In this dataset, random forest method predicts with very high accuracy levels gradient boosting method also predicts with reasonably good accuracy levels.