ML course Project

Introduction

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). 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.

Your submission for the Peer Review portion should consist of a link to a Github repo with your R markdown and compiled HTML describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5.

1. read and clean the data

NOTE: the raw testing data will become the VALIDATION Data

```
library (caret)

######## TRAINING DATA

trainData <- read.csv('./pml-training.csv', header=T)

# str(trainData) # not printed in the final report
# too many columns with "NA" data, they will be removed

# remove columns with near Zero variance from training data
nzvCol <- nearZeroVar(trainData)

cleanTnzv <- trainData[,-nzvCol]

# remove columns with all 'NA's in training data
cleanTrain<- cleanTnzv[ , colSums(is.na(cleanTnzv)) == 0]

# the first 7 column variable are not required for analysis

cleanTrain <- cleanTrain[,-c(1:7)]

######## VALIDATION (Quiz) DATA

validData <- read.csv('./pml-testing.csv', header=T)

# remove columns with near Zero variance from validation data
nzvCol <- nearZeroVar(validData)

cleanVnzv <- validData[,-nzvCol]

# remove columns with all 'NA's in validation data
cleanValid<- cleanVnzv[ , colSums(is.na(cleanVnzv)) == 0]

# the first 7 column variable are not required for analysis

cleanValid <- cleanValid[,-c(1:7)]
```

2. Partition the data into training and cross-valid(test) samples

The training data will be 75% of the data, randonly selected, the remainder will be cross-validation data. the testing data will remain separate for purpose of the Quiz.

```
set.seed(1234)

inTrain <- createDataPartition(y=cleanTrain$classe, p=3/4, list=FALSE)

training <- cleanTrain[inTrain,]

testing <-  cleanTrain[-inTrain,]

dim(training); dim(testing)
```

```
## [1] 14718    52
```

```
## [1] 4904   52
```

```
# [1] 14718  52
# [1] 4904   52
```

3. Modelling

Three differnt model algorithms were tried.

a. Decision trees with CART (rpart)
b. Random forest decision trees (rf)
c. Gradient Boosting Model (gbm)

The rf model in particluar to hours to run the firt time. I discovered the trControl funciton, to reduce computing time.

a. Decision trees with CART (rpart)

```
fitControl <- trainControl(method='cv', number = 3, verboseIter=FALSE)

# a. Decision trees with CART (rpart)

# create model with training data
model_RP <- train(classe~., data= training, method= "rpart")

# predict outcome of test data
predict_RP <- predict(model_RP, testing)

# compare test data prediction with classe
cm_RP <- confusionMatrix(predict_RP, testing$classe)

cm_RP$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   5.630098e-01   4.522395e-01   5.489913e-01   5.769532e-01   2.844617e-01
## AccuracyPValue  McnemarPValue
##   0.000000e+00  3.759720e-227
```

b. Random forest decision trees (rf)

```
model_RF <- train(classe~., data= training, method= "rf", trControl=fitControl)

# side note: because the model RF took hours to build the first time,
# I learned to saved/exported it to load anytime. Later added trControl.
# save(model_RF, file = "model_RF.RData")
# load(file = "model_RF.RData")

predict_RF <- predict(model_RF, testing)

cm_RF <- confusionMatrix(predict_RF, testing$classe)

cm_RF$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.9946982      0.9932928      0.9922412      0.9965339      0.2844617
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

c. Decision trees with Gradient Boosting Model (gbm)

```
model_GBM <- train(classe~., data= training, method= "gbm", trControl=fitControl, verbose=FALSE)

predict_GBM <- predict(model_GBM, testing)

cm_GBM <- confusionMatrix(predict_GBM, testing$classe)

cm_GBM$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   9.620718e-01   9.520033e-01   9.563413e-01   9.672434e-01   2.844617e-01
## AccuracyPValue  McnemarPValue
##   0.000000e+00   9.867397e-07
```

Apply selected model to Quiz 20 test samples provided

We have chosed the Random Forest model for it highest accuracy.

```
predict(model_RF, cleanValid)
```

```
##  [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
```