Peer-graded Assignment: Prediction Assignment Writeup

Purpose

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

Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Loading and preprocessing the data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Exploratory Analysis

Firstly load the data after downloading from the link provided.

Data import and clean up.

```
#if(!file.exists('dactivity.csv')){
#    unzip('activity.zip')
#}
data <- read.csv("data/pml-training.csv", na.strings=c("NA","#DIV/0!", ""))
final_test <- read.csv("data/pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))

 # Delete columns with all missing values
  data<-data[,colSums(is.na(data)) == 0]
  final_test <-final_test[,colSums(is.na(final_test)) == 0]

  # Delete variables are irrelevant to our current project: user_name, raw_timestamp_part_1, raw_timest
```

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

Now we have the data in data variable.

```
dim(data)
```

```
## [1] 19622    53
```

```
# split the training data into a sub sets.
inTrain <- createDataPartition(y=data$classe,p=0.7, list=FALSE)
training <- data[inTrain,]
testing <- data[-inTrain,]
```

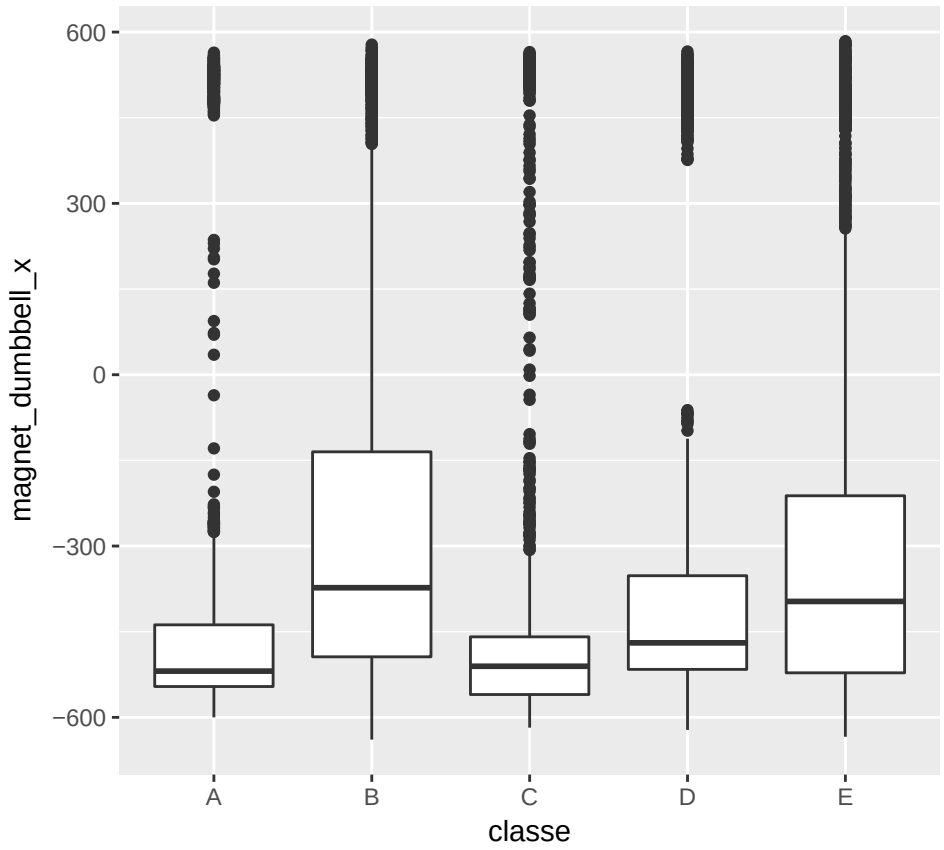
Using some basic tools we learn there are 160 variables over 19622 obervations on the training data set. This has been redued to 53 that will be used for for the cross validation. We also learn that the obervations appear to record at short intervals during the lifting of the barbell and then it seems the "new window" observation is the beginning of the next lift. This data row has a greater amount observations. This data is considered in the summary.

In order to understand the data better and density and box plot is created for comparing each classe.

```
if (!dir.exists('data/density')) {
  # for our purpose lets just assume if the one dir isn't there none of this has run
  dir.create('data/density', showWarnings = FALSE, recursive = TRUE)
  dir.create('data/boxplot', showWarnings = FALSE, recursive = TRUE)
  coln = colnames(data)
  coln
  for (i in coln) {
    str = paste("data/density/", i,".png", sep="")
    qplot(data[[i]],colour=classe,data=data,geom="density")
    ggsave(str)
    dev.off()
  }
  for (i in coln) {
    str = paste("data/boxplot/", i,".png", sep="")
    p <- ggplot(training, aes_string("classe", i))
    p + geom_boxplot()
    ggsave(str)
  }
}
```

By going through the slides it is easy to see that there are not many variables that are obvious good predictors.

```
    p <- ggplot(training, aes_string("classe", "magnet_dumbbell_x"))
    p + geom_boxplot()
```



One interesting boxplot chart showing the variable magnet_dumbbell_x which is included in the model. Selection creteria was based on observations with differing means. This was just an initial pass to start to understand the data.

After going through the data exporation process the follow variables are selected: magnet_dumbbell_x + accel_arm_x + gyros_arm_x + gyros_arm_y + roll_forearm + yaw_dumbbell + accel_belt_z + pitch_forearm.

```
modRF_org <- randomForest(classe ~ magnet_dumbbell_x + accel_arm_x + gyros_arm_x + gyros_arm_y + roll_f
            ,data=training)
```

Lets check out our exploration model.

```
pnbRF_org = predict(modRF_org,testing)
confusionMatrix(pnbRF_org, testing$classe)$overall['Accuracy']
```

```
##  Accuracy
## 0.9517417
```

So we see that with some very simple analysis we can get a pretty good model. about 95%

Cross Validation

Lets see what happens when we try some different types of prediction models.

Using rpart type prediction. Note I'm using all the variables and not th 8 from the initial selection. Using all the variables will get a higher accuarcy for all the models I've tried. However, I'm not including every aspect of the data exploration. Also, as I will highlight in the conclusion, these medthods do not cover the data within the "new window" type of observations.

```
model1 <- rpart(classe ~ ., data=training, method="class")
prediction1 <- predict(model1, testing, type = "class")
confusionMatrix(prediction1, testing$classe)$overall['Accuracy']
```

```
##  Accuracy
## 0.7388275
```

~74% accuaracy which is not nearly as good.

Support vector machine method

```
svmr <- svm(classe ~ magnet_dumbbell_x + accel_arm_x + gyros_arm_x + gyros_arm_x +roll_forearm + yaw_du
            ,data = training)
pSVM <- predict(svmr, testing)
  confusionMatrix(pSVM, testing$classe)$overall['Accuracy']
```

```
##  Accuracy
## 0.7315208
```

Using all variables gets better results.

```
svmr <- svm(classe ~ . ,data = training)
pSVM <- predict(svmr, testing)
  confusionMatrix(pSVM, testing$classe)$overall['Accuracy']
```

```
##  Accuracy
## 0.9420561
```

about 94%

Random forest with all 53 variables has highest accuracy of about ~99%

```
mod <- randomForest(classe ~ .  ,data=training)
pRF_all <- predict(mod, testing)
  confusionMatrix(pRF_all, testing$classe)$overall['Accuracy']
```

```
##  Accuracy
## 0.9940527
```

- Highest Accuracy ~99% *

Prediction on test data

Since random forest with the 53 variables has highest accurary will will use this model for final prediction and compare it to the random forest with only 8 variables.

```
rfPred <- predict(modRF_org, final_test)
svmrPred <- predict(svmr, final_test)
rfPredAll <- predict(mod, final_test)

prediction <- data.frame(cbind(rfPred, rfPredAll, svmrPred))

colnames(prediction) <- c("Random Forest 8 var", "Random Forest 53 var", "SVM 53 var")
```

```
knitr::kable(prediction)
```

Random Forest 8 var	Random Forest 53 var	SVM 53 var
2	2	2
1	1	1
2	2	1
1	1	1
1	1	1
5	5	5
4	4	4
2	2	2
1	1	1
1	1	1
2	2	2
3	3	3
2	2	2
1	1	1
5	5	5
5	5	5
1	1	1
2	2	2
2	2	2
2	2	2

Notice that 8 var's vs 53 gets the same results. which is interesting. All three models get same results, except SVM has one difference (see 3rd row).

K-folds out of sample error

```
k.folds <- function(k) {
    folds <- createFolds(training$classe, k = k, list = TRUE, returnTrain = FALSE)
    for (i in 1:k) {
        model <- modRF_org <- randomForest(classe ~ magnet_dumbbell_x + accel_arm_x + gyros_arm_x + gyr
            ,data=training)

        predictions <- predict(model, newdata = training[-folds[[i]],], type = "class")
        accuracies.dt <- c(accuracies.dt,
                           confusionMatrix(predictions, training[-folds[[i]], ]$classe)$overall[[1]])
    }
    accuracies.dt
}

accuracies.dt <- c()
accuracies.dt <- k.folds(5)
accuracies.dt
```

```
## [1] 1.000000 0.999818 0.999909 1.000000 0.999727
```

From the output above we see that the values are all very close to or equal to 1. So this is a very good model.

```
v <- c()
v <- replicate(5, k.folds(5))
accuracies.dt <- c()
for (i in 1 : 5) {
```

```
    accuracies.dt <- c(accuracies.dt, v[,i])
}

mean.accuracies <- mean(accuracies.dt)
lci <- mean(accuracies.dt) - sd(accuracies.dt) * 1.96
uci <- mean(accuracies.dt) + sd(accuracies.dt) * 1.96
```

From the output above we see that the out-of-sample error is likely between “r lci” and “r uci”.

Conclusions

During this data exploration we see that random forest seems to have the highest accuracy. So much so that even when using only 8 varaibles accuracy is still great than the next best model SVM. We also see that as the data test set becomes small, the need to use all 53 observations become less import as the model with only 8 preforms well in the out of sample testing.

Note about data with “new window”

Depending on such things as computational limitations and actual use case for these predictions, there may be simpler ways to understand how well someone is preforming an excersie.