Practical Machine Learning Course Project

The goal of this project was to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants and accurately predict how the lift was performed. In the data set this would be the ‘classe’ variable.

When loading in the data, I replaced all fields with ‘#DIV/0!’, blanks and NA’s to be imported as R’s default N/A value.

```
library(caret)
library(randomForest)
library(Hmisc)
library(doParallel)
library(foreach)

training <- read.csv("~/pml-training.csv",na.strings=c("#DIV/0!", "","NA") )
testing <- read.csv("pml-testing.csv", na.strings=c("#DIV/0!", "","NA") )
```

Data Manipulation

The original data set includes a low of missing data. My approach to this analysis is to use a complete data set. There is a lot of information in the data set even if we exclude the non-complete portions. I also removed columns that would not be useful in predictions, such as time-stamps, windows, and user names. I also split the training set in order for cross-validation.

```
#Limiting down variables to only complete data sets
#Remove columns with N/As
training <- training[,which(colSums(!is.na(training))==19622)]
testing <- testing[,which(colSums(!is.na(testing))==20)]

#user names and timestamps will not help in predicting performance
training <- training[,c(8:60)]
testing <- testing[,c(8:60)]

#Partition rows into training and crossvalidation
inTrain <- createDataPartition(training$classe, p = 0.6)[[1]]
crossv <- training[-inTrain,]
training <- training[ inTrain,]
```

Model Selection

I decided to cut down the training set dramatically in order to be able to use random forests on my computer. I had a lot of issues trying to run random forest models on the complete data set. In order to optimize run time of the random forest model. I only attempted 6 random forests with 40 trees each.

```
#Data cutdown for rf to process quickly
set.seed(12345)
training2 <- training[1:300,]
registerDoParallel()
x <- training[-ncol(training)]
y <- training$classe
rf <- foreach(ntree=rep(40, 6), .combine=randomForest::combine, .packages='randomForest') %dopar% {
randomForest(x, y, ntree=ntree)
}
```

Results

The random forest model's results on the training set is 100% accurate. The prediction results for the random forest model against the cross validation data set was 99.3% accurate. We'd expect an out of sample error to be within the 95% confidence interval of the prediction accuracy. Thus the out of sample error rate should be between 0.5% and 0.9%.

Given these results, I am satisfied with this random forest model especially given how it was optimized for run time on my computer by using only a small portion of the data set.

```
predictions1 <- predict(rf, newdata=training)
confusionMatrix(predictions1,training$classe)
```

```
## Warning: package 'e1071' was built under R version 3.1.1
```

```
##
## Attaching package: 'e1071'
##
## The following object is masked from 'package:Hmisc':
##
##     impute
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3348    0    0    0    0
##          B    0 2279    0    0    0
##          C    0    0 2054    0    0
##          D    0    0    0 1930    0
##          E    0    0    0    0 2165
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (1, 1)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    1.000    1.000    1.000    1.000
## Specificity             1.000    1.000    1.000    1.000    1.000
## Pos Pred Value          1.000    1.000    1.000    1.000    1.000
## Neg Pred Value          1.000    1.000    1.000    1.000    1.000
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.194    0.174    0.164    0.184
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       1.000    1.000    1.000    1.000    1.000
```

```
predictions2 <- predict(rf, newdata=crossv)
confusionMatrix(predictions2,crossv$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 2230    8    0    0    0
##          B    2 1506   11    0    0
##          C    0    4 1357    7    2
##          D    0    0    0 1279    2
##          E    0    0    0    0 1438
## 
## Overall Statistics
##                                          
##                Accuracy : 0.995          
##                  95% CI : (0.994, 0.997)
##     No Information Rate : 0.284          
##     P-Value [Acc > NIR] : <2e-16         
##                                          
##                   Kappa : 0.994          
##  Mcnemar's Test P-Value : NA             
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.999    0.992    0.992    0.995    0.997
## Specificity             0.999    0.998    0.998    1.000    1.000
## Pos Pred Value          0.996    0.991    0.991    0.998    1.000
## Neg Pred Value          1.000    0.998    0.998    0.999    0.999
## Prevalence              0.284    0.193    0.174    0.164    0.184
## Detection Rate          0.284    0.192    0.173    0.163    0.183
## Detection Prevalence    0.285    0.194    0.175    0.163    0.183
## Balanced Accuracy       0.999    0.995    0.995    0.997    0.999
```

Testing Set Predictions and Courersa Submission Code

Below is the submission results using the Coursera provided code.

```r
answers = predict(rf, newdata=testing)
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)
  }
}
pml_write_files(answers)
```