Machine Learning Course Project Report

Background

The purpose of this course project is 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://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

The participants performed the actions in the following ways:
Class A - performance exactly according to the specification
Class B - throwing the elbows to the front
Class C - lifting the dumbbell only halfway
Class D - lowering the dumbbell only halfway
Class E - throwing the hips to the front
Comparison among three models were perfomed to determine which best predicted the class of performance of these exercises given a training set of variables. The models were random forest (model 1), rpart(model 2), and knn - k-nearest neighbour classification (model 3).

Data Acquisition and Cleaning

The training and test data for this project were downloaded. The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.

```{r}
library(caret); library(rpart); library(rpart.plot); library(knitr); library(randomForest); library(rattle)
trainURL<-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
gradeURL<-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training<-read.csv(url(trainURL), header=TRUE, as.is = TRUE, stringsAsFactors=FALSE, sep=',',
na.strings=c('NA', '#DIV/0!', ''))
```

View the structure and clean the data.
The first 7 columns which did not contribute to the model and several variables containing NA's were removed. The same procedures were applied to the project test set which was renamed "grade" to preserve integrity until final model application. Near zero variance was used to check for remaining predictors that contained a single value. The training set was reduced to 52 variables.

```{r}
str(training[,100:160])
training$classe<-as.factor(training$classe)
trainingall<-training
cleanNA<-apply(trainingall,2,function(x) {sum(is.na(x))})
training2<-trainingall[,which(cleanNA==0)]
training<-training2[,8:60]
grade<-read.csv(url(gradeURL), header=TRUE, as.is = TRUE, stringsAsFactors=FALSE, sep=',',
na.strings=c('NA', '#DIV/0!', ''))
gradeall<-grade
grade2<-gradeall[,which(cleanNA==0)]
grade<-grade2[,8:60]
trainnzv<-nearZeroVar(training, saveMetrics=TRUE)
print(training)
```

Partition the Training Set

The training set was partitioned into a training and testing subset.

~~~
```{r}
set.seed(999)
inTrain<- createDataPartition(y=training$classe, p=.6, list=FALSE)
training<-training[inTrain,]
testing<-training[-inTrain,]
trainvars<-names(training)
```
~~~

Create the Models

Include cross validation in each of the models and compare the predictions among the models.

Model 1 - Random Forest

~~~
```{r}
validationgroup<-trainControl(method="cv", number = 5, allowParallel = TRUE, verboseIter =TRUE)
model1<-train(classe~., data=training, method="rf", trControl=validationgroup)
model1
predictionrf<-predict(model1,testing)
testing$predrfright<-predictionrf=testing$classe
table(predictionrf,testing$classe)
M1<-confusionMatrix(predictionrf, testing$classe)
````
~~~

Model 2 - rpart

~~~
```{r}
model2<-train(classe~., data=training, method="rpart", trControl=validationgroup)
predictrpart<-predict(model2, testing)
table(predictrpart,testing$classe)
M2<-confusionMatrix(predictrpart, testing$classe)
print(M2)
```
~~~

Model 3 -knn Nearest Neighbour Classification

~~~
```{r}
model3<-train(classe~., data=training, method="knn", trControl=validationgroup)
predictknn<-predict(model3, testing)
table(predictknn,testing$classe)
M3<-confusionMatrix(predictknn, testing$classe)
print(M3)
```
~~~

Compare the Models

When comparing the predictions with the class, it is clear that the random forest method best predicts the class with an accuracy of 100% and a 95% confidence interval of 0.999 to 1. Such results indicate the possibility of overfitting, and caution should be employed. See the model 1 prediction table.

~~~
```{r}
print(M1)
~~~

	Reference				
Prediction	A	B	C	D	E
A	1343	0	0	0	0
B	0	907	0	0	0
C	0	0	831	0	0
D	0	0	0	760	0
E	0	0	0	0	872

```Model 1 – random forest

Model 3, knn, predicts the class with an accuracy of 94.7% and a 95% confidence interval of 0.940 to 0.953.  The expected out of sample error for this model would be 7.9% and could be used as a prediction model for this dataset, but fails to predict as well as the random forest method.  For example, Class A was misclassified 39 times. See the model 3 table.

```{r}
print(M3)
```

Model 3 - KNN

	Reference				
Prediction	A	B	C	D	E
A	1315	24	9	6	9
B	4	839	16	4	19
C	9	19	790	36	15
D	13	18	10	711	20
E	2	7	6	3	809

Model 2, rpart, demonstrated the least accurate in predicting class with an accurace of 50.1% with a 95% confidence interval of 0.454 to 0.517. The expected out of sample error is 49.9%, that is, about essentially the accuracy of a coin flip. The partitioning placed most of the actions in classes A, B and C and only one in class E and none in class D. See the model 2 table.

```{r}
print(M2)
```
```

Model 2 - rpart

```
          Reference
Prediction    A    B    C    D    E

         A 1221  386  373  345  120

         B   19  240   12  139   42

         C   94  281  446  276  294

         D    0    0    0    0    0

         E    9    0    0    0  416
```

Given these results, the random forest method was applied to the graded test set for this project.

~~~
```{r}
forgrade<-predict(model1,grade)
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(forgrade)
```
~~~

When these results were evaluated within the course, the answers were in 100% agreement with the correct answers.