Practical Machine Learning - Project Report

My procedure to make predictions about the manner in which the participants did the excercise was a follows:

1. Load libraries

```
library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
library("randomForest")
```

```
## randomForest 4.6-7
## Type rfNews() to see new features/changes/bug fixes.
```

2. Read in training data

```
data <- read.table("pml-training.csv", na.strings=c("", " ", "NA"), header = TR
UE, sep = ",")
```

3. Remove columns that contains NA values (since most of the values in those columns are NAs)

```
data <- data[ ,colSums(is.na(data)) == 0]
```

4. Remove the first seven covariates that should not have any relationship with the outcome (i.e. user_names, date, etc)

```
head(data[,c(1:7)])
```

```
##   X user_name raw_timestamp_part_1 raw_timestamp_part_2   cvtd_timestamp
## 1 1  carlitos           1323084231               788290 05/12/2011 11:23
## 2 2  carlitos           1323084231               808298 05/12/2011 11:23
## 3 3  carlitos           1323084231               820366 05/12/2011 11:23
## 4 4  carlitos           1323084232               120339 05/12/2011 11:23
## 5 5  carlitos           1323084232               196328 05/12/2011 11:23
## 6 6  carlitos           1323084232               304277 05/12/2011 11:23
##   new_window num_window
## 1         no         11
## 2         no         11
## 3         no         11
## 4         no         12
## 5         no         12
## 6         no         12
```

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

5. Split the training dataset into a training and a testing data set for *Cross Validation* (60 % training and 40% testing) with the createDataPartition within the Caret Packeage. (The actual test data set with the 20 observations will be termed as validation data set, to avoid consuffusion)

```
inTrain <- createDataPartition(y=data$classe, p=0.6, list=FALSE)
training <- data[inTrain,]
testing <- data[-inTrain,]
```

6. Train model on the training data set using randomForest function with all defaults.

```
modelFit <- randomForest(classe ~ ., training)
```

7. Make predicions on the testing set

```
predictions <- predict(modelFit, newdata = testing)
```

8. Calculate the *Out of Sample Error* on the testing set (Cross Validation).

```
OOS_error <- confusionMatrix(predictions,testing$classe)
OOS_error
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2225   13    0    0    0
##          B    6 1500   12    0    0
##          C    0    5 1354   17    2
##          D    0    0    2 1268    4
##          E    1    0    0    1 1436
##
## Overall Statistics
##
##                Accuracy : 0.992
##                  95% CI : (0.9897, 0.9938)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9898
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9969   0.9881   0.9898   0.9860   0.9958
## Specificity            0.9977   0.9972   0.9963   0.9991   0.9997
## Pos Pred Value         0.9942   0.9881   0.9826   0.9953   0.9986
## Neg Pred Value         0.9988   0.9972   0.9978   0.9973   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2836   0.1912   0.1726   0.1616   0.1830
## Detection Prevalence   0.2852   0.1935   0.1756   0.1624   0.1833
## Balanced Accuracy      0.9973   0.9926   0.9930   0.9925   0.9978
```

9. As can be seen in the Confussion Matrix, just a few samples were miss classified. The acuracy of the prediction was 0.9945.

10. Finally, the Trained model was applied to predict the outcome (classe variable) of the 20 observations in the validation data set.

```
validate <- read.table("pml-testing.csv", na.strings=c("", " ", "NA"), header =
 TRUE, sep = ",")
validate_clean <- validate[ , colSums(is.na(data)) == 0]
validate_clean <- validate_clean[,-c(1:7)]
answer <- predict(modelFit, validate_clean)
```