Practical Machine Learning Final Report: Exercise Prediction

I have downloaded the file, reading it then creating partition.

Reading and creating partition in the File

```
library(caret)
```

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

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

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rpart)
train_in <- read.csv("pml-training.csv")
validation <- read.csv('pml-testing.csv', header=T)
set.seed(127)
training_sample <- createDataPartition(y=train_in$classe, p=0.7, list=FALSE)
training <- train_in[training_sample, ]
testing <- train_in[-training_sample, ]
dim(training)
```

```
## [1] 13737   160
```

```
dim(testing)
```

```
## [1] 5885  160
```

Removing near 0 variables and then removing variable

```
NZV <- nearZeroVar(training)
training <- training[, -NZV]
testing  <- testing[, -NZV]
dim(training)
```

```
## [1] 13737   109
```

```
dim(testing)
```

```
## [1] 5885  109
```

```
#Now removing variables that are mostly NA
AllNA    <- sapply(training, function(x) mean(is.na(x))) > 0.95
training <- training[, AllNA==FALSE]
testing  <- testing[, AllNA==FALSE]
dim(training)
```

```
## [1] 13737    59
```

```
dim(testing)
```

```
## [1] 5885   59
```

Model building

The three model types I'm going to test are:

1.Random forest decision trees (rf) 2.Decision trees with CART (rpart) 3.Method: Generalized Boosted Model

1. Random forest decision trees (rf)

```
 controlRF <- trainControl(method="cv", number=3, verboseIter=FALSE)
    modFitRandForest <- train(classe ~ ., data=training, method="rf",trControl=controlRF)
    predictRandForest <- predict(modFitRandForest, newdata=testing)
    confMatRandForest <- confusionMatrix(predictRandForest, testing$classe)
    confMatRandForest
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    0 1139    0    0    0
##          C    0    0 1026    0    0
##          D    0    0    0  964    0
##          E    0    0    0    0 1082
## 
## Overall Statistics
## 
##                Accuracy : 1
##                  95% CI : (0.9994, 1)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

2. Decision trees

```
modFitDecTree <- rpart(classe ~ ., data=training, method="class")
predictDecTree <- predict(modFitDecTree, newdata=testing, type="class")
confMatDecTree <- confusionMatrix(predictDecTree, testing$classe)
confMatDecTree
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    0    0    0    0
##          B    1 1138    0    0    0
##          C    0    1 1025    0    0
##          D    0    0    1  964    0
##          E    0    0    0    0 1082
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9995          
##                  95% CI : (0.9985, 0.9999)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9994          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9991   0.9990   1.0000   1.0000
## Specificity            1.0000   0.9998   0.9998   0.9998   1.0000
## Pos Pred Value         1.0000   0.9991   0.9990   0.9990   1.0000
## Neg Pred Value         0.9998   0.9998   0.9998   1.0000   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1934   0.1742   0.1638   0.1839
## Detection Prevalence   0.2843   0.1935   0.1743   0.1640   0.1839
## Balanced Accuracy      0.9997   0.9995   0.9994   0.9999   1.0000
```

3. Method: Generalized Boosted Model

```
controlGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
modFitGBM  <- train(classe ~ ., data=training, method = "gbm",trControl = controlGBM, verbose = FALSE)
```

```
## Loading required package: gbm
```

```
## Loading required package: survival
```

```
## 
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
## 
##     cluster
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.1
```

```
## Loading required package: plyr
```

```
modFitGBM$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 50 iterations were performed.
## There were 80 predictors of which 1 had non-zero influence.
```

```
predictGBM <- predict(modFitGBM, newdata=testing)
confMatGBM <- confusionMatrix(predictGBM, testing$classe)
confMatGBM
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    0    0    0    0
##          B    1 1138    0    0    0
##          C    0    1 1025    0    0
##          D    0    0    1  964    0
##          E    0    0    0    0 1082
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9995          
##                  95% CI : (0.9985, 0.9999)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.9994          
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9991   0.9990   1.0000   1.0000
## Specificity            1.0000   0.9998   0.9998   0.9998   1.0000
## Pos Pred Value         1.0000   0.9991   0.9990   0.9990   1.0000
## Neg Pred Value         0.9998   0.9998   0.9998   1.0000   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1934   0.1742   0.1638   0.1839
## Detection Prevalence   0.2843   0.1935   0.1743   0.1640   0.1839
## Balanced Accuracy      0.9997   0.9995   0.9994   0.9999   1.0000
```

Model Assessment (Out of sample error)

The accuracy of the 3 regression modeling methods above are:

Random Forest : 1 Decision Tree : 0.9995 GBM : 0.9995 In that case, the Random Forest model will be applied to predict the 20 quiz results (testing dataset) as shown below.

```
predictTEST <- predict(modFitRandForest, newdata=validation)
```